Intermec
Technologies Corporation

6910 Integrated and 6910 Telnet Gateway/Access Points

USER'S GUIDE

P/N 961-047-122
Revision A
September 1998

" **NOTICE**

The information contained herein is proprietary and is provided solely for the purpose of allowing customers to operate and service Intermec manufactured equipment and is not to be released, reproduced, or used for any other purpose without written permission of Intermec.

We welcome your comments concerning this publication. Although every effort has been made to keep it free of errors, some may occur. When reporting a specific problem, please describe it briefly and include the book title and part number, as well as the paragraph or figure number and the page number.

Send your comments to:
Intermec Technologies Corporation
Publications Department
550 Second Street SE
Cedar Rapids, IA 52401

This publication printed on recycled paper.

Acknowledgments

Portions of this product contain software which is licensed from and is copyrighted by Epilogue Technology Corporation, 1988–1995, all rights reserved.

DECnet and *VT* are registered trademarks of Digital Equipment Corporation.

Hewlett-Packard and *HP* are registered trademarks and *HP OpenView* is a trademark of Hewlett-Packard Company.

PC AT is a registered trademark of International Business Machines Corporation.

PROCOMM and *PROCOMM PLUS* are registered trademarks of DataStorm Technologies, Inc.

FCC Computer Compliance

" **NOTICE**

This equipment meets Class B digital device limits per Part 15 of FCC Rules. These limits protect against interference in a residential area. It emits, uses, and can radiate radio frequency energy. If you do not install and use the equipment according to its instructions, it may interfere with radio signals. However, there is no guarantee that interference will not occur in a particular installation.

If this equipment does cause harmful interference to radio or television reception, which can be determined by turning our equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the radio or television receiving antenna.
- Increase the separation between the computer equipment and receiver.
- Connect the equipment into an outlet on a circuit different from that to which the radio or television receiver is connected.
- Consult the dealer or an experienced radio or television technician for help.

FCC Spread Spectrum Radio Certification

" NOTICE

This device is certified to operate under Part 15, Subpart C, Section 15.247 of the FCC rules for Intentional Radiation Products. This certification includes Docket 87-389 covering rules effective June 1994. It may not cause interference to authorized radio communication devices, and must accept any interference caused by those devices.

Antenna Requirements

" NOTICE

FCC rules section 15.203 and Canada's RSS-210 require that this device be operated using an antenna furnished by Intermec Technologies Corporation. The antenna coupling on this product has been designed to accept only antennas manufactured by us. Use of an antenna other than that furnished with the equipment is prohibited by FCC and Industry Canada rules.

Canadian Computer Compliance

This Class B digital apparatus meets all requirements of the Canadian Interference-Causing Equipment Regulations.

Cet appareil numerique de la classe B respecte toutes les exigences du Reglement sur le material boilleur du Canada.

Canadian Spread Spectrum Radio Certification

" NOTICE

This device complies with RSS-210 of Industry Canada. Operation is subject to the following two conditions: (1) this device may not cause interference, and (2) this device must accept any interference, including interference that may cause undesired operation of the device.

Canadian 2.4 GHz Radio License

" NOTICE

This device requires a radio license, unless it is installed totally inside a building. (Users must obtain this license)

Une licence radio est requise pour ces dispositifs, sauf pour ceux installés tout à fait à l'intérieur d'un bâtiment. (Il faut que l'utilisateur obtienne cette licence.)

Telephone Installation Warning Notices

The following notices apply to equipment that may be connected to telephone lines or systems. For your personal safety, and to protect this equipment from potential electrical or physical damage, do NOT connect equipment to telephone lines or data communication equipment unless the following warnings have been read, understood, and complied with.

- Never install telephone wiring during a lightning storm.
- Never install telephone jacks in wet locations unless the jack is specifically designed for wet locations.
- Never touch uninsulated telephone wires or terminals unless the telephone line has been disconnected at the network interface.
- Use caution when installing or modifying telephone lines.
- Avoid using telephone (other than cordless type) during an electrical storm. There may be a remote risk of electric shock from lightning.
- Do not use the telephone to report a gas leak in the vicinity of the leak.

Installation du téléphone : avertissements

Les avertissements qui suivent s'appliquent à tout équipement qui peut être branché aux lignes ou systèmes téléphoniques. Pour votre sécurité personnelle et pour protéger l'équipement de tout dommage électrique ou physique potentiel, NE PAS brancher un ordinateur tablette électronique ou ses périphériques aux lignes téléphoniques ou équipements avant que les avertissements suivants aient été lus, compris et observés :

- Ne jamais installer de câblage téléphonique pendant un orage électrique.
- Ne jamais installer de prise téléphonique dans un endroit humide à moins que la prise ait été spécifiquement conçue pour être utilisée dans les endroits humides.
- Ne jamais toucher les fils de téléphone ou de l'équipement terminal non isolés à moins que la ligne téléphonique n'ait été débranchée de l'interface réseau.
- User de prudence lors de l'installation ou de la modification de lignes téléphoniques.
- Éviter d'utiliser un téléphone (autre qu'un appareil téléphonique sans fil) pendant un orage électrique. Il pourrait y avoir un faible risque d'électrocution par la foudre.
- Ne pas utiliser le téléphone afin de signaler une fuite de gaz à proximité de la fuite.

B CAUTION: **Intermec Technologies Corporation suggests you buy cables from us to connect with other devices. Our cables are safe, meet FCC rules, and suit our products. Other cables may not be tested. They may cause problems from electrostatic discharge or induced energy. Our warranties do not cover loss, injury, or damage from other cables.**

CONTENTS

SECTION 1

SECTION 2

SECTION 3

APPENDIX A

FIGURES - 6910 INTEGRATED GATEWAY/ACCESS POINT

FIGURES - 6910 TELNET GATEWAY/ACCESS POINT

TABLES

Section 1

Introduction

Purpose of This Guide

The 6910 Integrated Gateway/Access Point and 6910 Telnet Gateway/Access Point are optional software loads for the 6710 Access Point. This user’s guide is a supplement to the *6710 Access Point User’s Guide* and covers only the differences between the gateway/access point and the 6710 Access Point. The following pages summarize the differences.

6910 Integrated Gateway/Access Point

Functionality: The 6910 Integrated Gateway/Access Point is a functional 6710 Access Point that also provides a serial controller function. Gateway/access point functionality is equivalent to the INTERMEC^{R} RC4030E Gateway for small wireless station populations.

Local access: The procedure to access the gateway/access point’s system software through the DIAG (HOST) port differs from the procedure for the 6710 Access Point. The devices also use different DIAG port cables.

Configuration: The gateway/access point’s system software contains gateway and host configuration options not found on the 6710 Access Point.

Section 2 describes integrated gateway/access point operation. It also describes how to connect the gateway/access point to a host, access the system software, and configure gateway and host options.

6910 Telnet Gateway/Access Point

Functionality: The 6910 Telnet Gateway/Access Point is a functional 6710 Access Point with additional terminal emulation gateway functionality. The device supports VT220, Telnet 3270, and Telnet 5250 over Ethernet. Gateway/access point functionality is equivalent to the INTERMEC 6950 Enterprise Gateway Server for small wireless station populations.

Run-time license: The gateway/access point requires a run-time license file for normal operation. The 6710 Access Point does not require a license file.

Configuration: The gateway/access point's system software contains gateway and host configuration options not found on the 6710 Access Point.

Section 3 describes Telnet gateway/access point operation. It also describes how to obtain the run-time license file and configure gateway and host options.

INCA

Norand Corporation is now part of Intermec Technologies Corporation. As part of our continuing efforts to offer the broadest range of system solutions in the industry, the 6910 Integrated Gateway/Access Point, 6910 Telnet Gateway/Access Point, and other open wireless LAN components have been merged into the INTERMEC Integrated Network Communications Architecture (INCA).

Where appropriate, we have continued to use the Norand name in references to the open wireless LAN to maintain continuity with existing product in the field.

Related Publications

The following publications are available. They include information about hardware and software products related to or used with the gateway/access points and the networks on which they operate. Contact your Sales Representative for ordering information.

2100 UAP User's Manual

The manual for the 2100 Universal Access Point (UAP) describes how to install, configure, and troubleshoot this access point. The manual's part number (P/N) is 067150.

6710 Access Point User's Guide

The user's guide for the 6710 Access Point covers these areas: features and functionality, installation, system configuration, software download, and troubleshooting. It also contains specifications for the access point and WLIF, 900 MHz, and S-UHF radios. The manual's part number is 961-047-081.

Wireless Station User's Guides

User's guides for wireless stations describe how to set up, operate, and maintain the terminals.

Specific wireless station guides are:

- ***1100 Series Data Terminal User's Guide*** **(P/N 961-047-069)**
- ***5900 Series User's Guide*** **(P/N 961-047-121)**
- ***PEN*KEY^R Model 6400 Hand-Held Computer User's Guide*** **(P/N 961-047-093)**
- ***PEN*KEY Model 6500/6550 User's Guide*** **(P/N 961-047-099)**
- ***RT1700 Radio Data Terminal User's Guide*** **(P/N 961-047-068)**

Programmer's Reference Guides

3270 Terminal Emulation Programmer's Reference Guide (P/N 977-047-040)

This guide describes how terminal emulation stations emulate IBM 3278 Model 2 terminal operation through the 3270 data stream. This guide also covers asynchronous controller commands, and terminal emulation station commands and orders.

5250 Terminal Emulation Programmer's Reference Guide (P/N 977-047-039)

This guide describes how terminal emulation stations emulate IBM 5291 Display Station operation through the 5250 data stream. This guide also covers 5250 display data stream commands.

Native Terminal Emulation Asynchronous Programmer's Reference Guide (P/N 977-047-038)

This guide describes components in the radio network using asynchronous NORAND Native communications. This guide also contains commands and orders terminal emulation stations can accept from a host.

VT220/ANSI Terminal Emulation Programmer's Reference Guide (P/N 977-047-037)

This guide describes how terminal emulation stations emulate VT220 terminal operation. This guide also describes VT220 received codes, transmitted codes, and character sets.

Section 2

6910 Integrated Gateway/Access Point

Features and Functional Overview

" **NOTE:** *Refer to Section 2, “Features and Function Overview,” in the 6710 Access Point User’s Guide for information about access point bridging functionality, configuration, network management, and hardware components.*

The 6910 Integrated Gateway/Access Point combines the functionality of a gateway device and the 6710 Access Point to support the NORAND R Native communications type for small installations. As an optional wired bridge, the gateway/access point bridges frames between the wired Ethernet LAN and wireless stations on the radio network. An optional function is to serve as the connection point for several types of wireless stations, which include terminal emulation stations and PC-compatible computers.

When configured with host options, the gateway/access point picks up data frames from the wireless stations. It translates the frames into the appropriate host protocol and sends the data to the host through its DIAG (HOST) port.

This section describes how to set host and gateway configuration options. For more information about the Native communications type, refer to the *Native Terminal Emulation Asynchronous Programmer’s Reference Guide* (P/N 977-047-038).

Wireless Station Support

The gateway/access point with the WLIF or 900 MHz radio option supports these wireless stations:

- RT1100, RT1700, and RT5900 series radio terminals
- PEN*KEY® 6400, 6500, and 6550 Computers

The S-UHF radio supports these wireless stations:

- RT5921 Mobile Mount Radio Terminal
- TM1100 Radio Terminal with RM31 (integrated scanning) and RM11 radio modules
- TM1700 Radio Terminal with RM31 (integrated scanning) and RM11 radio modules

Configuration of individual radio options is discussed in the *6710 Access Point User's Guide.*

Host Connectivity

Figure 2-1 is a sample configuration with host connectivity.

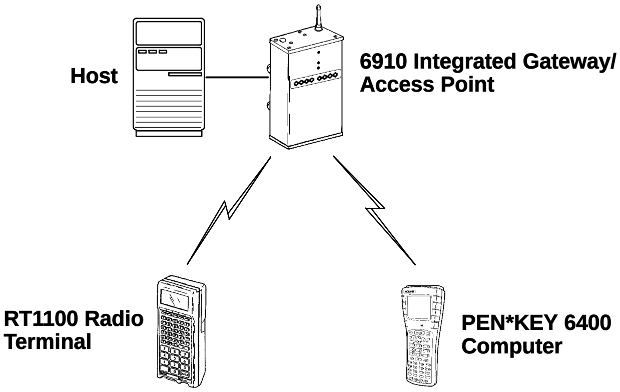


Figure 2-1
Sample Host Connectivity Configuration

Ethernet Connectivity

Figure 2-2 is a sample configuration with Ethernet connectivity. The figure also shows 6710 Access Points, which can provide additional coverage and wireless links to secondary Ethernet LANs.

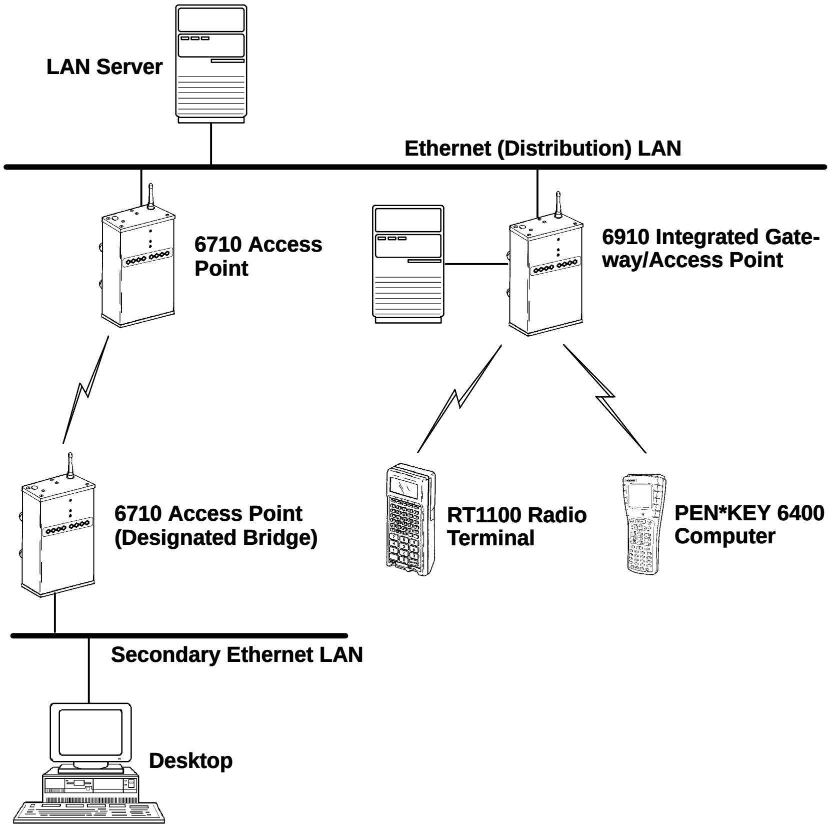

LAN Server
Ethernet (Distribution) LAN
6710 Access Point
6910 Integrated Gate-way/Access Point
6710 Access Point (Designated Bridge)
RT1100 Radio Terminal
PEN*KEY 6400 Computer
Secondary Ethernet LAN
Desktop

Figure 2-2
Sample Ethernet Connectivity Configuration

Configuration of secondary Ethernet LANs and wireless links is discussed in the *6710 Access Point User's Guide.*

Installation

Refer to Section 3, "Installation," in the *6710 Access Point User's Guide* for information about the following:

- Preparing for the installation.
- Collecting the proper Ethernet components and communication equipment. Note that the gateway/access point and 6710 Access Point use different DIAG port cables. See "Local DIAG (HOST) Port Access" below for the appropriate cables for the gateway/access point.
- Finding the best location.
- Mounting the access point.
- Connecting the access point to Ethernet media.
- Installing WLIF, 900 MHz, and S-UHF PC cards.
- Applying power to the access point.

Local DIAG (HOST) Port Access

The gateway/access point's DIAG (HOST) port is a 9-pin D-sub communication port that communicates at RS-232 levels. Use this port to configure the gateway/access point, download new software, retrieve statistics, and connect to the host computer.

The following chart lists the cables that connect a PC to the gateway/access point's DIAG (HOST) port. Cable pin-outs start on page 2-21.

PC Port	**Cable P/N**
9-pin male	226-208-00X*
25-pin male	226-208-00X* *plus* the 9-pin to 25-pin adapter provided with the cable
25-pin female	226-243-00X*

** -001 = 10 feet, -002 = 25 feet, -003 = 50 feet*

Connecting to the Host

To connect the gateway/access point to the host, see Figure 2-3 and the procedure following it.

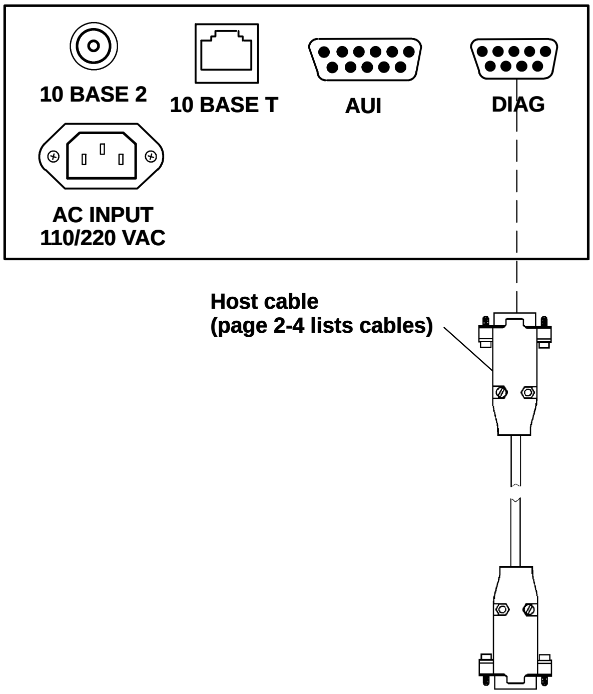


Figure 2-3
Host Connection

1. Plug the 9-pin male connector on the host cable into the DIAG (HOST) port.
2. Plug the connector on the other end of the cable into the designated communications port on the host.

Configuration

The following pages describe how to create a local DIAG (HOST) port session with the gateway/access point and access FLASH and ROM. The pages also describe host and gateway options and how to configure them. Refer to Section 4, “Configuration,” in the *6710 Access Point User’s Guide* for information about how to do the following:

- Create a Telnet session and Web browser session.
- Set up TCP/IP, bridge, and security options. However, see page 2-19 in this section for specific gateway/access point configuration guidelines.

Refer to Section 5, “Software Download,” in the *6710 Access Point User’s Guide* for information about the software download process, including the file system structure, File Menu commands, and the ROM command monitor. When referring to Section 5, substitute USTART29.BIN (the FLASH file for the 6710 Access Point) with IGAP6910.BIN (the FLASH file for the integrated gateway/access point).

Creating a Local DIAG Port Session

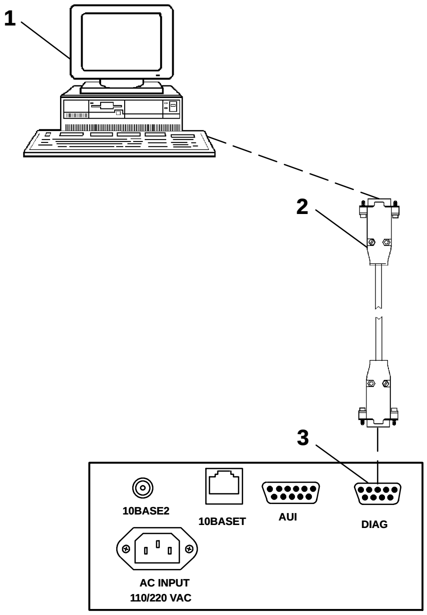
DIAG port functionality is shared on the gateway/access point as follows:

- When the gateway/access point initially powers up, its DIAG port functions as the ROM command interpreter. ROM DIAG port default settings are **9600, N81**.
- As the device continues to power up, the DIAG port becomes a host port with default settings of **9600, E71**.

The functional DIAG port and host port settings are independent and must be configured separately.

You establish a local DIAG (HOST) port session with the gateway/access point through a VT100 terminal emulation program, such as PROCOMM PLUS installed on a PC. When the program is running you issue host command "CMT27,2" to put the DIAG (HOST) port into configuration mode. When you exit the configuration menus, the port automatically switches into host mode.

To create a local session, see Figure 2-4 and the procedure following it. You should carefully review the procedure first to become familiar with the process.



1. **PC with terminal emulation program**
2. **Cable: P/N 226-208-00X for a 9-pin male PC COM port (a 9-pin to 25-pin adapter is required for a 25-pin port)**
   or
   Cable: P/N 226-243-00X for a 25-pin female PC COM port
3. **6910 Integrated Gateway/Access Point DIAG (HOST) port**

Figure 2-4
Local DIAG (HOST) Port Session

1. Ensure the terminal emulation program is installed on the PC.
2. With both the PC and gateway/access point powered OFF, connect the communication cable to the appropriate PC COM port.
3. Connect the other end of the communication cable to the DIAG port on the gateway/access point. Turn the PC on.
4. After the PC boots, start the terminal emulation program.
5. Set the terminal emulation program's options according to what you want to do: Access the configuration menus, or access the ROM command monitor.

Accessing the Configuration Menus

1. Set the terminal emulation parameters. If you are configuring this gateway/access point for the first time, set the parameters to the default settings for FLASH mode:

   9600, **E71**, **full duplex**

   If you have already changed the default settings, set the parameters to those you set in FLASH mode through the configuration menus.
2. Plug the gateway/access point into the outlet. These messages appear (the first two are ROM power-up messages and the third is FLASH):

```
Press any key within 5 seconds to enter ROM command monitor
Executing file IGAP6910.BIN from segment <segment number>
POWER UPMUX V9.22,16 (this is the default host port message)
```

3. Do not press a key within 5 seconds of the first ROM message. Wait until the FLASH (third) message displays.

   The FLASH message displays only at the default setting of 9600 E71 or at the speed you set in FLASH mode through the configuration menus. If you do not see the FLASH message, you have selected the wrong baud rate in the serial communications program.

4. After the FLASH message displays, enter this command to put the gateway/access point's DIAG (HOST) port into configuration mode:

   CMT27,2

   Note the following:

   - Type CMT in uppercase and do not enter spaces between any characters in the command.
   - There is a 1 second timeout period between characters. If you wait more than 1 second to type the next character, the gateway/access point will not recognize the command.
   - If you type the wrong character you cannot use the backspace key to correct it. You must either wait 1 second or press [Enter] before trying again.

Accessing the ROM Command Monitor

▪ **NOTE:** *If you are replacing an RM2216, RM3216, or RC3250 Controller, see "Notice for UHF Controllers" on page 2-10.*

1. Set the terminal emulation parameters. If you are configuring this device for the first time, set the parameters to the default settings for ROM mode:

   9600, **N81**, **full duplex**

   If you have already changed the default settings, set the parameters to those you set in ROM mode through the ROM command monitor.

2. Plug the gateway/access point into the outlet. These messages appear (the first two are ROM power-up messages and the third is FLASH):

Press any key within 5 seconds to enter ROM command monitor
Executing file IGAP6910.BIN from segment <segment number>
POWER UPMUX V9.22,16 (this is the default host port message)

3. Press any key within 5 seconds of the first ROM message.

   The two ROM messages display at the default setting of 9600 N81 or at the speed you set in ROM mode.

   Note that if the gateway/access point is in Power-Up Quiet mode (versus Power-up Normal mode, the default setting), the ROM messages do not display.

4. Refer to Section 5, “Configuration,” in the *6710 Access Point User’s Guide* for more information about the ROM command monitor, PQ mode, and PN mode.

Notice for UHF Controllers

If you are replacing an RM2216, RM3216, or RC3250 Controller with the integrated gateway/access point, applications that were running on the controller can become confused when the ROM power-up messages appear on the network because the host is expecting standard controller power-up messages.

Do **one** of the following to suppress the ROM power-up messages:

- Access the ROM command monitor and turn Power-up Quiet mode to ON. Reboot the gateway/access point.
- Plug the gateway/access point into the outlet *without the host connected to the gateway/access points' DIAG (HOST) port.* After both ROM power-up messages display, connect the host to the gateway/access point.

Gateway Default and Site Settings

The gateway/access point is factory configured with the default gateway settings listed in the following chart. You may need to change the defaults to match the way your system is set up. You can record your site's settings in the table for reference.

Gateway Option	Default	Site Setting
Name	HOST	
Gateway.Compression	Disabled	
Host Type	Async	
[Async Parms]		
Baud Rate	9600	
Stop Bits	1	
Parity	Even	
Data Bits	7	
Host Timeout	0	
Address Type	Old	
Enhanced R Polling	Disabled	
Multiple Buffering	Disabled	

Gateway Options

Options for the gateway application are as follows:

```
Name                      “HOST”
Gateway.Compression       <Disabled>
Host Type                 <Async>
[Async Parms]
```

" **NOTE:** *[Async Parms] appears only if Host Type is set to Async.*

Name

The Name option is an arbitrary name that identifies this gateway/access point. Wireless stations use the arbitrary name to connect to this gateway/access point. Note that Name is **not** your application’s host server name.

The setting for Name must match the host name specified for each wireless station communicating with the host through this gateway/access point. The names must match so that the wireless stations can communicate with the host. Refer to the wireless station’s user’s guide for more information about specifying host names.

The setting for Name is **case sensitive**. For example, if the host name specified for the wireless stations is typed in all uppercase, the name for the gateway/access point must also be typed in all uppercase. Figure 2-5 shows an example of how host name HOST1 is set on a gateway/access point and wireless station.

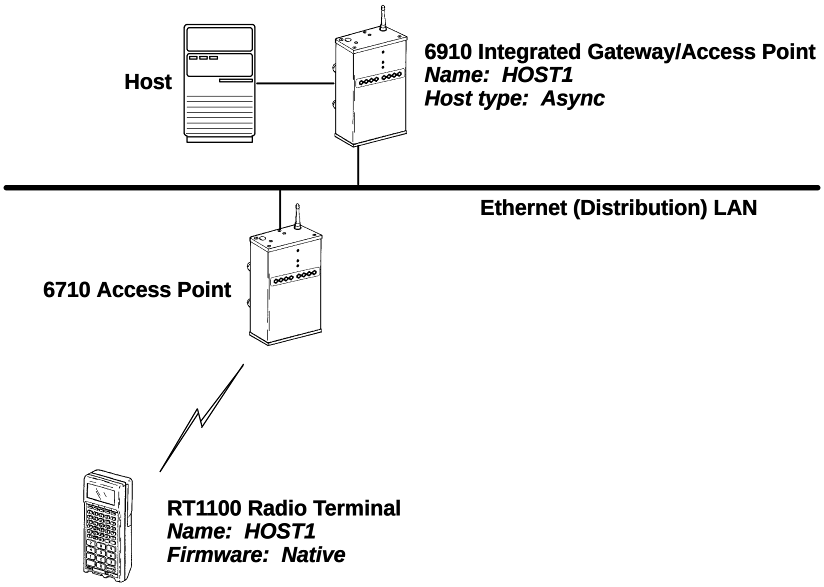


Figure 2-5
Host Names

The prompt for Name is:

```
Range is:
16 chars
```

Type 16 or fewer alphanumeric characters for the name. The program inserts quotation marks around your entry.

Gateway.Compression

Gateway.Compression uses a general data compression algorithm to reduce the size of data being sent to a wireless station. Settings are:

Enabled
Disabled

Setting	Description
Enabled	Compresses most outbound data from the gateway/access point to the wireless stations. Inbound data sent from the station is not compressed. Compression takes place on a client-by-client basis as allowed by each client.
Disabled *(default)*	Does not compress data.

On most systems, compression reduces the RF (radio frequency) load by compressing most of the outbound data from the host to the wireless station. It is strongly recommended that you enable Gateway.Compression.

Compression does not significantly increase wireless station counts or supported transaction rates. It does improve response time consistency in applications where the same data or screens are frequently repeated. The effectiveness of compression varies with the size and number of unique screens.

While compression may reduce the amount of data being sent through the RF system, it increases the gateway/access point's processing load. Therefore, it may be beneficial to disable compression for some systems.

Host Type

Use Host Type to select the type of communications the host uses:

```
                    Async
                    None
```

Setting	Description
Async *(default)*	Indicates that one host connects through RS-232 and uses a proprietary protocol to multiplex all wireless stations. Use Async for NORAND Native communications and ADK.
None	Disables gateway/access point functionality. The device functions only as a 6710 Access Point.

[Async Parms]

Options for [Async Parms] configure RS-232 parameters and the asynchronous multiplexing protocol:

```
        Baud Rate               <9600>
        Stop Bits               <1>
        Parity                  <Even>
        Data Bits               <7>
        Host Timeout            0
        Address Type            <Old>
        Enhanced R Polling      <Disabled>
        Multiple Buffering      <Disabled>
```

Baud Rate

Baud Rate sets the serial communication speed in bits per second. Settings are:

300
1200
2400
4800
9600
19200
38400

The default is 9600. To avoid confusion, the Baud Rate setting should match the serial baud rate setting for the ROM command monitor. Different baud rate settings, however, do not affect gateway/access point operation. Refer to Section 5, "Software Download," in the *6710 Access Point User's Guide* for more information about the serial baud rate setting for the ROM command monitor.

Stop Bits

Stop Bits sets the number of stop bits to be sent with each character:

1
2

The default is 1.

Parity

Parity sets the data parity:

```
None
Even
Odd
```

The default is Even.

Data Bits

Data Bits sets the number of data bits per character:

```
7
8
```

The default is 7.

Host Timeout

Host Timeout is the number of seconds of host inactivity before the host is considered down. The prompt is:

```
Range is:
0..255
```

The default of zero means the gateway/access point never times out the host. Ten or more seconds is recommended. Do not set Host Timeout to zero if the wireless station is configured to communicate with up to three hosts through the wireless station's Advanced Setup firmware option.

After the host timeout value has been exceeded, the wireless station displays “Host is Down.” The user can then reboot the wireless station. The gateway/access point tries to connect with the next host defined for the wireless station.

Address Type

For each frame sent to the host, the originating station can be fixed at one or three characters. Settings are:

```
Normal
Old
```

Setting	**Description**
Normal	Wireless station addresses consist of three octets.
Old *(default)*	Gateway/access point is backward compatible to the RM2216 series of multiplexers. Wireless station addresses consist of one octet instead of three, and gateway/access point responses are in the MUX response format.

Enhanced R Polling

Enhanced R Polling determines how the gateway/access point responds to host commands with data (if available). Settings are:

```
Enabled
Disabled
```

Setting	**Description**
Enabled	Gateway/access point responds to host commands with data without sending acknowledgments (the wireless station response is the acknowledgment). This reduces the number of polls necessary to be generated from the host.
Disabled *(default)*	Gateway/access point responds to host commands with data by sending acknowledgments.

Multiple Buffering

Multiple Buffering determines how the gateway/access point stores messages. Settings are:

```
Enabled
Disabled
```

Setting	Description
Enabled	Gateway/access point stores more than one wireless station message from the host while it sends previous messages to the wireless station.
Disabled *(default)*	Gateway/access point stores only one wireless station message from the host.

Configuration Guidelines

Refer to Section 4, “Configuration,” in the *6710 Access Point User’s Guide* for standard access point configuration guidelines. The following pages provide specific guidelines for the integrated gateway/access point.

IP Address

Manually assign an IP address to the gateway/access point and disable DHCP operation. DHCP and the IP address are configurable through the [Tcpip] option on the Main Options Menu.

LAN ID

For terminal emulation, a single gateway/access point can support a mixed population of wireless stations regardless of which LAN ID approach you choose. On the open wireless LAN, the gateway/access point and wireless stations establish sessions based on the gateway/access point's name and terminal number.

The LAN ID does not bind wireless stations and gateway/access points. To use a single gateway/access point to support a mixed population of radios, ensure that only one wireless station has a given host name and terminal number combination.

OWL/IP

Ensure the gateway/access point will not be a designated bridge for an OWL/IP tunnel. This is accomplished by not entering the gateway/access point's IP address for the OWL/IP [IP Addresses] configuration option.

Root Priority

Configure the root priority as follows:

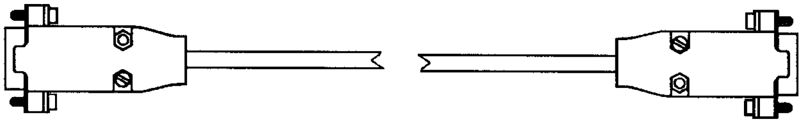
- If the integrated gateway/access point is the only access point on the network (that is, no 6710 Access Points are installed), set its root priority to a number between 1 and 7 (inclusive). Do not set its root priority to zero.
- In larger installations with multiple 6710 Access Points on the backbone, set the integrated gateway/access point's root priority to zero so it is prohibited from becoming the super root.

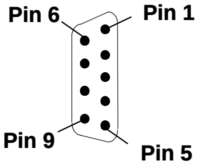
The root priority is configurable through the [Bridge] option on the Main Options Menu.

DIAG (HOST) Port Cables

DIAG (HOST) Port to 9-pin Male Host Port

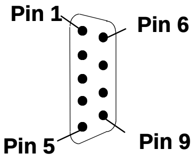
P/N: 226-208-001 (10 feet)
226-208-002 (25 feet)
226-208-003 (50 feet)





J-1
9-pin, D-sub
Female

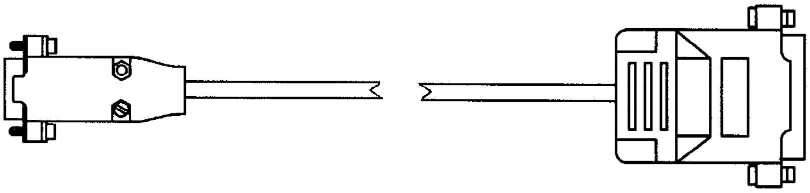
J2-7	jumper	J2-8
J1-7	jumper	J1-8
J1-6	brown	J2-4
J1-5	green	J2-5
J1-4	red	J2-6
J1-3	yellow	J2-2
J1-2	black	J2-3
J1-HSG	shield	J2-HSG
J1-HSG	drain	J2-HSG

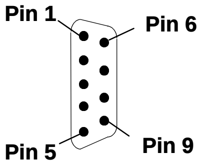


J-2
9-pin, D-sub
Female

DIAG (HOST) Port
to 25-pin Female Host Port

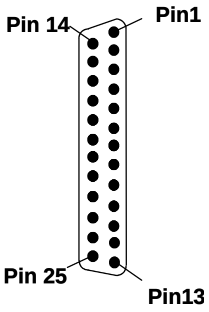
P/N: 226-243-001 (10 feet)
226-243-002 (25 feet)
226-243-003 (50 feet)





J-2
9-pin, D-sub
Female

J1-HSG	-	J2-HSG	SHIELD
J1-HSG	-	J2-HSG	DRAIN
J2-7	-	J2-8	-
J1-4	-	J1-5	-
J1-20	DSR	J2-6	DSR
J1-7	GND	J2-5	GND
J1-6	DTR	J2-4	DTR
J1-3	RXD	J2-3	RXD
J1-2	TXD	J2-2	TXD



J-1
25-pin, D-sub
Male

Section 3

6910 Telnet Gateway/Access Point

Features and Functional Overview

" **NOTE:** *Refer to Section 2, "Features and Function Overview," in the 6710 Access Point User's Guide for information about access point bridging functionality, configuration, network management, and hardware components.*

The 6910 Telnet Gateway/Access Point is a functional access point running TCP to WTP (Wireless Transport Protocol) gateway software. The gateway/access point enables 16 or fewer wireless stations to establish a Telnet connection with 8 or fewer hosts. Terminal emulation support is limited to TNVT, TN5250, and TN3270 on the products listed on page 2-2 in Section 2, "6910 Integrated Gateway/Access Point."

As an optional wired bridge, the gateway/access point bridges frames between the wired Ethernet LAN and wireless stations on the radio network. An optional function is to serve as the connection point for several types of wireless stations, which include terminal emulation stations and PC-compatible computers.

When configured with host options, the gateway/access point receives NORAND Network Layer (NNL) frames from the wireless stations. It translates the frames into the appropriate host protocol (TNVT, TN3270, or TN5250) and sends the data to the host over the Ethernet physical media.

This section describes how to set gateway and host configuration options. For more information about 3270, 5250, and VT220 communications, refer to the following programmer's guides:

- *3270 Terminal Emulation Programmer's Reference Guide* (P/N 977-047-040)
- *5250 Terminal Emulation Programmer's Reference Guide* (P/N 977-047-039)
- *VT220/ANSI Terminal Emulation Programmer's Reference* (P/N 977-047-037)

VT220 Character Mode Emulation

VT220 is a character mode emulation. Each keystroke entered into the wireless station may be transmitted as a separate radio frequency (RF) packet. Normally, host systems are configured for remote echo, which returns an RF packet to the wireless station for each keystroke. Scanning data from the wireless station and screen data from the host are sent as blocks.

Where possible, Intermec recommends that you use VT220 block mode or VT340-based forms to improve VT performance.

Sample Network Configuration

Figure 3-1 shows a sample network configuration. Note that the gateway/access point directly connects to the Ethernet medium.

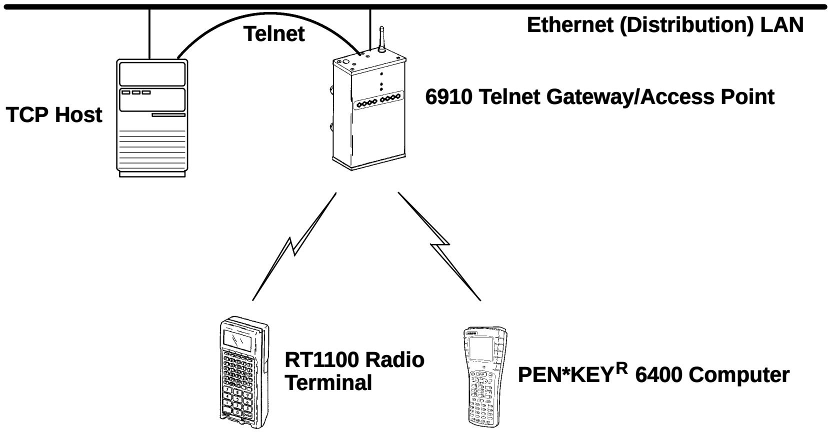

Ethernet (Distribution) LAN
Telnet
TCP Host
6910 Telnet Gateway/Access Point
RT1100 Radio Terminal
PEN*KEY^R 6400 Computer

Figure 3-1
Sample Network Configuration

Run-Time Licensing

A run-time licensing capability is in effect for the Telnet gateway/access point. Licensing information is contained in a plain text file (SECURITY.INI) located in the gateway/access point’s active data segment.

The file is protected by a digital signature algorithm that appends a signature to the file. The system software verifies whether a valid signature exists, and either uses or rejects the license file. The plain text approach allows license information to be read. Tampering or modification corrupts the signature.

Following is a sample license file:

```
[Identification]
Serial=0002121533
#
[Features]
TcpGateway=True
#
[Keys]
Key1=fde4d44d36f62eb89e6ec887c990956b
```

Telnet Gateway/Access Point Operation

The Telnet gateway/access point requires a license file with a valid signature to operate as an access point with gateway functionality. If an invalid license file or no license file is present, the device does the following:

- If an invalid file is present, the device displays system message “TCP Gateway Disabled” when it powers up. It then operates only as an access point (gateway functionality is disabled), and reboots every 15 minutes. This delay allows remote dial-in to diagnose the file and add a license file if necessary.
- If no file is present, the device operates only as an access point; gateway functionality is disabled. No rebooting occurs.

Obtaining the Run-Time License File

When you order a Telnet gateway/access point from the factory, the device is shipped with the run-time license file installed in its active data segment.

You can also enable a 6710 Access Point as a Telnet gateway/access point. In this case, contact your Sales Representative for information on how to obtain the run-time license file.

Installation

Refer to Section 3, “Installation,” in the *6710 Access Point User’s Guide* for information about the following:

- Preparing for the installation.
- Collecting the proper Ethernet components and communication equipment. Note that the Telnet gateway/access point and 6710 Access Point use the same DIAG port cables.
- Mounting the access point.
- Connecting the access point to Ethernet media.
- Installing WLIF, 900 MHz, and S-UHF PC cards.
- Applying power to the access point.
- Finding the best location.

" **NOTE:** *The gateway/access point should reside on the same subnet as the 6710 Access Point operating as the super root.*

Configuration

The following pages describe gateway and host options and how to configure them. Refer to Section 4, “Configuration,” in the *6710 Access Point User’s Guide* for information about how to do the following:

- Create a local DIAG port session, Telnet session, and Web browser session.
- Access FLASH and ROM. The procedure for accessing the gateway/access point’s FLASH and ROM is the same as for the 6710 Access Point.
- Configure TCP/IP, bridge, and security options. However, see page 3-14 in this section for specific gateway/access point configuration guidelines.

Refer to Section 5, “Software Download,” in the *6710 Access Point User’s Guide* for information about the software download process, including the file system structure, File Menu commands, and the ROM command monitor.

When referring to Section 5, substitute USTART29.BIN (the FLASH file for the 6710 Access Point) with TGAP6910.BIN (the FLASH file for the Telnet gateway/access point).

Gateway Default and Site Settings

The gateway/access point is factory configured with the default gateway settings listed in the following chart. You may need to change the defaults to match the way your system is set up. You can record your site’s settings in the table for reference.

Gateway Option	**Default**	**Site Setting**
Name	HOST	
Gateway.Compression	Disabled	
Host Type	TCP	
[TCP Parms]		
IP Address	0.0.0.0	
TCP Port	23	
Idle Timeout	Disabled	
Lost Timeout	Disabled	

Gateway Options

Options for the gateway application are as follows:

```
Name                      "HOST"
Gateway.Compression       <Disabled>
Host Type                 <TCP>
[TCP Parms]
```

" **NOTE:** *[TCP Parms] appears only if Host Type is set to TCP.*

Name

The Name option is an arbitrary name that identifies this gateway/access point. Wireless stations use the name to connect to this gateway/access point. Note that Name is **not** your application's host server name.

The setting for Name must match the host name specified for each wireless station communicating with the host through this gateway/access point. The names must match so that the wireless stations can communicate with the host. Refer to the wireless station's user's guide for more information about specifying host names.

The setting for Name is **case sensitive**. For example, if the host name specified for the wireless stations is typed in all uppercase, the name for the gateway/access point must also be typed in all uppercase. Figure 3-2 shows an example of how host name HOST1 is set on a gateway/access point and wireless station.

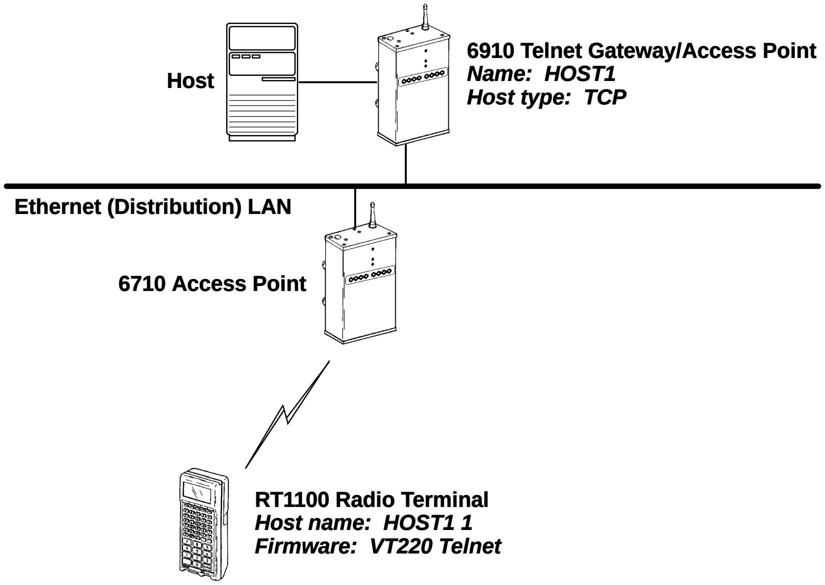


Figure 3-2
Host Names

The prompt for Name is:

```
Range is:
16 chars
```

Type 16 or fewer alphanumeric characters for the name. The program inserts quotation marks around your entry.

Gateway.Compression

Gateway.Compression uses a general data compression algorithm to reduce the size of data being sent to a wireless station. Settings are:

Enabled
Disabled

Setting	Description
Enabled	Compresses most outbound data from the gateway/access point to the wireless stations. Inbound data sent from the station is not compressed. Compression takes place on a client-by-client basis as allowed by each client.
Disabled *(default)*	Does not compress data.

On most systems, compression reduces the RF (radio frequency) load by compressing most of the outbound data from the host to the wireless station. It is strongly recommended that you enable Gateway.Compression.

Compression does not significantly increase wireless station counts or supported transaction rates. It does improve response time consistency in applications where the same data or screens are frequently repeated. The effectiveness of compression varies with the size and number of unique screens.

While compression may reduce the amount of data being sent through the RF system, it increases the gateway/access point's processing load. Therefore, it may be beneficial to disable compression for some systems.

Host Type

Use Host Type to select the type of communications the host uses. Settings are:

```
                    TCP
                    None
```

Setting	Description
TCP *(default)*	Enables you to configure one to eight hosts on the network. See [TCP Parms].
None	Disables gateway functionality. The device functions only as a 6710 Access Point.

[TCP Parms]

Use [TCP Parms] to configure one to eight hosts on the network. Options are:

```
    IP Address    TCP Port   Idle Timeout    Lost Timeout
 1  0.0.0.0             23   <Disabled>      <Disabled>
 2  0.0.0.0             23   <Disabled>      <Disabled>
 3  0.0.0.0             23   <Disabled>      <Disabled>
 4  0.0.0.0             23   <Disabled>      <Disabled>
 5  0.0.0.0             23   <Disabled>      <Disabled>
 6  0.0.0.0             23   <Disabled>      <Disabled>
 7  0.0.0.0             23   <Disabled>      <Disabled>
 8  0.0.0.0             23   <Disabled>      <Disabled>
```

The number in the first column is an arbitrary number you assign to a specific host on the network. Select a number and press [Enter] to configure the host options, as follows:

```
IP Address        0.0.0.0
TCP Port          23
Idle Timeout      <Disabled>
Lost Timeout      <Disabled>
```

IP Address

This option is the IP address of the Nth (first through eighth) host configuration. The prompt is:

```
Range is:
4 nums 0..255
```

The default is 0.0.0.0, which you can use to disable this configuration.

EXAMPLE: Figure 3-3 shows an example with two host configurations.

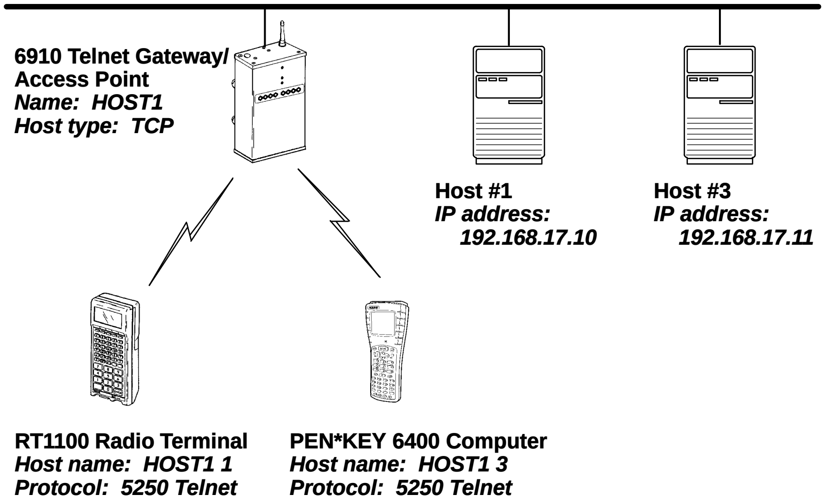


Figure 3-3
Sample Configuration With Two Hosts

In Figure 3-3, the IP addresses for [TCP Parms] are configured as follows:

```
   IP Address      TCP Port   Idle Timeout Lost Timeout
1  192.168.17.10      23      <Disabled>   <Disabled>
2  0.0.0.0            23      <Disabled>   <Disabled>
3  192.168.17.11      23      <Disabled>   <Disabled>
4  0.0.0.0            23      <Disabled>   <Disabled>
5  0.0.0.0            23      <Disabled>   <Disabled>
6  0.0.0.0            23      <Disabled>   <Disabled>
7  0.0.0.0            23      <Disabled>   <Disabled>
8  0.0.0.0            23      <Disabled>   <Disabled>
```

In this example, the RT1100 Radio Terminal connects to host #1 when its firmware is configured with host name HOST1 1, where:

- “HOST1” is the setting for the gateway/access point’s Name option (described on page 3-7).
- The second “1” in the host name is the arbitrary host number for the [TCP Parms] option (described on page 3-10).

The PEN*KEY 6400 Computer connects to host #3 when its firmware is configured with host name HOST1 3, where:

- “HOST1” is the setting for the gateway/access point’s Name option.
- The “3” in the host name is the arbitrary host number.

Refer to the wireless station’s user’s guide for more information about setting its host name and protocol option through the firmware configuration menus.

TCP Port

This setting is the TCP port number of the Nth host configuration. The prompt is:

```
Range is:
0..65535
```

The default is 23, the common Telnet port. Change this number if the host provides the service on another port or if the wireless station needs to connect to a different service.

Idle Timeout

Idle Timeout indicates how long a wireless station connected to a host remains connected without communicating. Settings are:

```
Disabled
5 Minutes
15 Minutes
30 Minutes
1 Hour
2 Hours
4 Hours
8 Hours
16 Hours
```

The default is Disabled.

When Idle Timeout expires, the gateway/access point closes the session. Some hosts periodically send data to a client to see if it is still connected. This activity may prevent Idle Timeout from expiring.

Lost Timeout

Lost Timeout indicates how long a wireless station connected to the Nth host is allowed to remain out of range before the session closes. Settings are:

Disabled
5 Minutes
15 Minutes
30 Minutes
1 Hour
2 Hours
4 Hours
8 Hours
16 Hours

The default is Disabled.

Lost Timeout is useful for closing old sessions when a wireless station has been taken out of service. If you set the time too short, a wireless station that is temporarily out of radio range prematurely loses the connection.

Configuration Guidelines

Refer to Section 4, “Configuration,” in the *6710 Access Point User’s Guide* for standard access point configuration guidelines. The following pages provide specific guidelines for the Telnet gateway/access point.

IP Address

Manually assign an IP address to the gateway/access point and disable DHCP operation. DHCP and the IP address are configurable through the [Tcpip] option on the Main Options Menu.

LAN ID

For terminal emulation, a single gateway/access point can support a mixed population of wireless stations regardless of which LAN ID approach you choose. On the open wireless LAN, the gateway/access point and wireless stations establish sessions based on the gateway/access point's name and terminal number.

The LAN ID does not bind wireless stations and gateway/access points. To use a single gateway/access point to support a mixed population of radios, ensure that only one wireless station has a given host name and terminal number combination.

OWL/IP

Ensure the gateway/access point will not be a designated bridge for an OWL/IP tunnel. This is accomplished by not entering the gateway/access point's IP address for the OWL/IP [IP Addresses] configuration option.

Root Priority

Configure the root priority as follows:

- If the Telnet gateway/access point is the only access point on the network (that is, no 6710 Access Points are installed), set its root priority to a number between 1 and 7 (inclusive). Do not set its root priority to zero.
- In larger installations with multiple 6710 Access Points on the backbone, set the Telnet gateway/access point's root priority to zero so it is prohibited from becoming the super root.

The root priority is configurable through the [Bridge] option on the Main Options Menu.

Appendix A

MIB

Product Contents

The 6910 Integrated Gateway/Access Point MIB is on disk part number 215-895-001. The 6910 Telnet Gateway/Access Point MIB is on disk part number 215-896-001. Order the appropriate MIB through your Sales Representative.

The following products are available for management of the open wireless LAN/INCA LAN:

- HP OpenView for Windows
- OWLView for HP OpenView for UNIX
- OWLView for HP OpenView for Windows

About This Product

Gateway/access point MIBs are packaged to provide basic network management capability for the open wireless LAN. The 6910 Integrated Gateway/Access Point maintains the following management objects, which are specific to its operation:

- **6910IMIB.MIB** — contains all of the Intermec management objects supported on the 6910 Integrated Gateway/Access Point.
- **RFC1213.MIB** — is the standard MIB-II.
- **RFC1398.MIB** — is the standard Ethernet MIB.

These MIBs are on the gateway/access point's MIB disk. You need to load the MIBs onto your management platform to query the device for these management objects.

The 6910 Telnet Gateway/Access Point maintains **6910TMIB.MIB**, **RFC1213.MIB**, and **RFC1398.MIB**. These MIBs are on the gateway/access point's MIB disk.

Getting Started

Install the MIBs onto your management system *in this order:*

1. **RFC1213.MIB**
2. **RFC1398.MIB**
3. **6910IMIB.MIB** *or* **6910TMIB.MIB**

" **NOTE:** *If you are using HP OpenView for Windows, use the OpenView Control/SNMP Manager/Manage Database menu items to add the previous listed MIBs to the HP OpenView MIB database. If you are* ***not*** *using HP OpenView for Windows, consult your network management station user's guide for instructions on adding MIBs.*

MIB-II Information

MIB-II is for use with network management protocols in TCP/IP-based internets. The 6910 Integrated and 6910 Telnet Gateway/Access Points support most of MIB-II. Table A-1 contains the major groups.

Table A-1
MIB-II Information

MIB Family	OID	Purpose	Groups
System	1.3.6.1.2.1.1	Model and device type	
Interfaces	1.3.6.1.2.1.2	I/O ports	
AT	1.3.6.1.2.1.3	Table of IP to MAC/DLC address	
IP	1.3.6.1.2.1.4	IP process	
ICMP	1.3.6.1.2.1.5	ICMP process	
TCP	1.3.6.1.2.1.6	TCP process	
UDP	1.3.6.1.2.1.7	UDP process	
EGP *	1.3.6.1.2.1.8	EGP process	
CMOT *	1.3.6.1.2.1.9	Historical inclusion for OSI support	
Transmission	1.3.6.1.2.1.10	Allows for data based on I/O port type	dot3 (Ethernet)
SNMP	1.3.6.1.2.1.11	Allows data to be collected about SNMP devices	

* *The 6910 Integrated and 6910 Telnet Gateway/Access Points do not support EGP and CMOT.*

Gateway/Access Point MIB Information

Intermec has structured its proprietary management information similar to MIB-II. In addition to MIB-II, the gateway/access points support information specific to their operation.

EXAMPLE: Device system information is found under `nSystem`, similar to MIB-II System. The OID for the nSystem group ends in "1," just as the OID for MIB-II system ends in "1."

Table A-2 shows gateway/access point MIB information.

Table A-2
MIB Information

MIB Family	OID	Purpose	Groups
nSystem	1.3.6.1.4.1.469.1000.2.1	Model, device type, software, file system	hw, file, fsinfo, segment, dir, sysErrors, criticalErrors
nInterfaces	1.3.6.1.4.1.469.1000.2.2	I/O ports	nifx, portState, portStats, ptxq, pmsg
nSNMP	1.3.6.1.4.1.469.1000.2.11	SNMP	community, trapTarget
nBridge	1.3.6.1.4.1.469.1000.2.17	Bridging	rt, brg, addr, brgState, bridgeStats
nTransport	1.3.6.1.4.1.469.1000.2.102	Transport	wst, hlit, wcbst
nApplication	1.3.6.1.4.1.469.1000.2.104	Device applications	gw Session (gs, rcbs, hd, td, ahost, tgate*)
nControl	1.3.6.1.4.1.469.1000.2.105	Device control	powerUp, softwareDownLoad

* *The tgate MIB group applies only to the 6910 Telnet Gateway/Access Point.*

Access to Management Information

Access to Intermec management information is obtained with the proper COMMUNITY name. Intermec provides three levels of access. This table outlines the levels with the required community name.

" **NOTE:** *Community strings are case-sensitive.*

Community String	Access Type	Description of Access Type
public	READ-ONLY	May read MIB objects, but not write or change values. **EXCLUSIONS:** Will not be able to read or write the Community Table.
CR52401	READ-WRITE	May read MIB objects. May write to MIB objects that have read-write access. **EXCLUSIONS:** Will not be able to read or write the Community Table.
secret	SUPER-USER	May read MIB objects. May write to MIB objects that have read-write access. Can read and write the Community Table.

The names of the community strings for each community or access group are stored in (norand.manage.norandNet.nSNMP.v1Config.communityTable). These three records may be viewed and modified if used with the SUPER-USER community. There is a maximum, allowing for three levels of access.

Records may be added or deleted via setting the communityStatus object to enable, disable, or delete. The first row in the CommunityTable is reserved for the SUPER-USER community definition. This record is not removable. This is a fixed record to ensure read-write access to the MIBs on the gateway/access point. Note the communityName for the first record can be changed to ensure end-user control of security for the gateway/access point.

MIB-II Notes

System Group — Three fields in the MIB-II system group are writable. Those fields are: sysContact, sysName, and sysLocation. It is important that these values be preserved in case the gateway/access point is powered (off and on) or rebooted. The following lists the number of characters for each field that will be preserved in the event of a power (off and on) or reboot.

sysContact:	31 characters
sysName:	31 characters
sysLocation:	39 characters

Interfaces Group — The ifTable.ifAdminStatus object is read-write accessible. However, this functionality has not been enabled.

IP Forwarding — IP Forwarding is disabled for this release of the gateway/access point. Therefore, the MIB-II ipForwarding object is not changeable.

MIB Directory

The following pages describe the various groups that the 6910 Integrated and 6910 Telnet Gateway/Access Points support. Table A-3 lists groups, their meaning, and page numbers where each group's table summary and definitions appear.

Table A-3
MIB Directory

		Group	MIB
Group	**Meaning**	**Summary**	**Definition**
Product OIDs			
products	INTERMEC^R Products	A-9	A-34
System Information			
hw	Hardware Information	A-10	A-34
fsinfo	File System Information	A-11	A-35
segment	File Segment Information	A-11	A-36
dir	Software Directory Listing	A-12	A-38
criticalErrors	Critical Errors Information	A-12	A-40
Interface Information			
nifx	Norand Extensions to Interfaces Table	A-13	A-41
portState	Port State Information	A-14	A-45
portStats	Port Statistics	A-15	A-49
ptxq	Port Transmit Queue	A-16	A-54
pmsg	Pending Message Services	A-17	A-57
SNMP Version 1 Configuration			
community	Community Table	A-18	A-59
trapTarget	Trap Target Table	A-18	A-61

Table A-3 (Continued)
MIB Directory

Group	Meaning	Group Summary	MIB Definition
Bridging Parameters			
rt	Route Table	A-19	A-63
brg	Bridge Table	A-20	A-66
addr	Address Table	A-21	A-68
brgState	Bridge State Information	A-21	A-70
bridgeStats	Bridge Statistics	A-23	A-74
Transport Groups			
wst	Wireless Transport Protocol (WTP) Status Table	A-24	A-76
hlit	High Level Interface Table	A-25	A-82
wcbst	WTP Control Block Status Table	A-26	A-83
Application Layer Groups			
gs	Gateway Statistics	A-27	A-88
rcbs	Radio Terminal Control Block Statistics	A-29	A-94
hd	Host Delay Statistics	A-30	A-100
td	Terminal Transaction Delay Statistics	A-30	A-102
ahost	Asynchronous Host application	A-31	A-104
tgate*	Gateway Connection Table	A-32	A-107
Control Groups			
powerUp	Power Up Objects	A-33	A-110
softwareDownLoad	Software Download	A-33	A-110

* *Applies only to the 6910 Telnet Gateway/Access Point.*

MIB Outline

Product OIDs

This group contains an Object IDentification (OID) for each INTERMEC device.

Table A-4
products GROUP

Device Products
norand.manage.products.x
(1.3.6.1.4.1.469.1000.1.x)

OID	Object Name	Object Type	Access
1	ap6710	OBJECT ID	Not Applicable (N/A)
2	gw4030	OBJECT ID	(N/A)
3	wnas	OBJECT ID	(N/A)
4	ts6950	OBJECT ID	(N/A)
5	gwap6910	OBJECT ID	(N/A)
6	uap2100	OBJECT ID	(N/A)
7	msd6710	OBJECT ID	(N/A)

System Information

The following groups contain system level objects describing hardware and file system configuration properties. The groups also contain information about critical errors.

" **NOTE:** *The MIB definition for each group starts on the page given below.*

" hw	Hardware Information (page A-34)
" fsinfo	File System Information (page A-35)
" segment	File Segment Information (page A-36)
" dir	Software Directory Listing (page A-38)
" criticalErrors	Critical Errors Information (page A-40)

Table A-5
hw GROUP

Device Hardware Information
norand.manage.norandNet.nSystem.hw.x
(1.3.6.1.4.1.469.1000.2.1.1.x)

OID	Object Name	Object Type	Access
1	hwPartNo	INTEGER	read
2	hwDescription	DisplayString	read
3	hwRevision	INTEGER	read
4	hwSerialNo	INTEGER	read
5	hwID	INTEGER	read

Table A-6
fsinfo GROUP

Device File System Information
norand.manage.norandNet.nSystem.file.fsinfo.x
(1.3.6.1.4.1.469.1000.2.1.3.1.x)

OID	Object Name	Object Type	Access
1	fsEnabled	INTEGER	read
2	fsMaxSectors	INTEGER	read
3	fsSectorSize	INTEGER	read
4	fsNumSegments	INTEGER	read
5	fsNumFiles	Gauge	read
6	fsBootSegment	INTEGER	read
7	fsDataSegment	INTEGER	read

Table A-7
segment GROUP

Device File Segment Information
norand.manage.norandNet.nSystem.file.segment.x
(1.3.6.1.4.1.469.1000.2.1.3.2.x)

OID	Object Name	Object Type	Access
2.1.1	segID	INTEGER	read
2.1.2	segFirstSector	INTEGER	read
2.1.3	segLastSector	INTEGER	read
2.1.4	segStatus	INTEGER	read
2.1.5	segSize	INTEGER	read
2.1.6	segFree	INTEGER	read

Table A-8
dir GROUP

Device Software Directory Listing
norand.manage.norandNet.nSystem.file.dir.x
(1.3.6.1.4.1.469.1000.2.1.3.3.x)

OID	Object Name	Object Type	Access
2.1.1	dirIndex	INTEGER	read
2.1.2	dirName	DisplayString	read
2.1.3	dirSegment	INTEGER	read
2.1.4	dirType	INTEGER	read
2.1.5	dirSize	INTEGER	read
2.1.6	dirDate	DisplayString	read
2.1.7	dirTime	DisplayString	read
2.1.8	dirVersion	DisplayString	read

Table A-9
criticalErrors GROUP

Device Critical Errors Information
norand.manage.norandNet.nSystem.sysErrors.criticalErrors.x
(1.3.6.1.4.1.469.1000.2.1.4.1.x)

OID	Object Name	Object Type	Access
1	ceEnabled	INTEGER	read
2	ceOverflow	INTEGER	read
3	ceReset	INTEGER	write
4.1.1	ceLogErrorCode	INTEGER	read
4.1.2	ceLogErrorCount	Counter	read

Interface Information

The following groups relate information about Norand interfaces, port state, port statistics, port transmit queue, and pending message services.

" **NOTE:** *The MIB definition for each group starts on the page given below.*

" nifx	Norand Extensions to Interfaces Table (page A-41)
" portState	Port State Information (page A-45)
" portStats	Port Statistics (page A-49)
" ptxq	Port Transmit Queue (page A-54)
" pmsg	Pending Message Services (page A-57)

Table A-10
nifx GROUP

Norand Extensions to MIB-II Interfaces Table
norand.manage.norandNet.nInterfaces.nifx.x
(1.3.6.1.4.1.469.1000.2.2.2.x)

OID	**Object Name**	**Object Type**	**Access**
4.1.1	nifxIndex	INTEGER	read
4.1.2	nifxType	INTEGER	read
4.1.3	nifxInDisabledDiscards	Counter	read
4.1.4	nifxInOverruns	Counter	read
4.1.5	nifxInHWOverruns	Counter	read
4.1.6	nifxInUcastDPkts	Counter	read
4.1.7	nifxInNUcastDPkts	Counter	read
4.1.8	nifxInLenErrors	Counter	read
4.1.9	nifxExcessiveDeferrals	Counter	read
4.1.10	nifxInNetIDDiscards	Counter	read
4.1.11	nifxInFragDiscards	Counter	read
4.1.12	nifxInUFilterDiscards	Counter	read
4.1.13	nifxInNUFilterDiscards	Counter	read
4.1.14	nifxInQFullDiscards	Counter	read

Table A-11
portState GROUP

Device Port State Information
norand.manage.norandNet.nInterfaces.portState.x
(1.3.6.1.4.1.469.1000.2.2.3.x)

OID	Object Name	Object Type	Access
4.1.1	psPort	INTEGER	read
4.1.2	psIfIndex	INTEGER	read
4.1.3	psAddress	PhysAddress	read
4.1.4	psType	INTEGER	read
4.1.5	psState	INTEGER	read
4.1.6	psCost	INTEGER	read
4.1.7	psHelloPeriod	INTEGER	read
4.1.8	psHelloCount	Counter	read
4.1.9	psMacdWindow	INTEGER	read
4.1.10	psMacdQSize	Gauge	read
4.1.11	psMacdTimeouts	Counter	read
4.1.12	psIsPrimary	INTEGER	read
4.1.13	psIsSecondary	INTEGER	read
4.1.14	psIsSecondaryCandidate	INTEGER	read
4.1.15	psSecondaryUniFlooding	INTEGER	read
4.1.16	psSecondaryMultiFlooding	INTEGER	read
4.1.17	psIsRadio	INTEGER	read
4.1.18	psPendEnabled	INTEGER	read

Table A-12

portStats GROUP

Device Port Statistics
norand.manage.norandNet.nInterfaces.portStats.x
(1.3.6.1.4.1.469.1000.2.2.4.x)

OID	Object Name	Object Type	Access
4.1.1	pstcPort	INTEGER	read
4.1.2	pstcInOWLPkts	Counter	read
4.1.3	pstcInUcastOWLDataPkts	Counter	read
4.1.4	pstcInNUcastOWLDataPkts	Counter	read
4.1.5	pstcInOWLErrors	Counter	read
4.1.6	pstcOutOWLPkts	Counter	read
4.1.7	pstcOutUcastOWLDataPkts	Counter	read
4.1.8	pstcOutNUcastOWLDataPkts	Counter	read
4.1.9	pstcOutOWLErrors	Counter	read
4.1.10	pstcParentLinkErrors	Counter	read
4.1.11	pstcAlertLinkErrors	Counter	read
4.1.12	pstcInUcastRelayPkts	Counter	read
4.1.13	pstcInNUcastRelayPkts	Counter	read
4.1.14	pstcOutUcastRelayPkts	Counter	read
4.1.15	pstcOutNUcastRelayPkts	Counter	read
4.1.16	pstcInUcastInbound	Counter	read
4.1.17	pstcInUcastOutbound	Counter	read
4.1.18	pstcInUcastSec	Counter	read
4.1.19	pstcInUcastFlood	Counter	read
4.1.20	pstcUcastDiscards	Counter	read
4.1.21	pstcInNUcastDiscards	Counter	read
4.1.22	pstcInUcastToIFC	Counter	read
4.1.23	pstcInNUcastToIFC	Counter	read
4.1.24	pstcOutDelayDiscards	Counter	read

Table A-13

ptxq GROUP

Device Port Transmit Queue
norand.manage.norandNet.nInterfaces.ptxq.x
(1.3.6.1.4.1.469.1000.2.2.5.x)

OID	Object Name	Object Type	Access
1.1.1	ptxqPort	INTEGER	read
1.1.2	ptxqRegQSize	Gauge	read
1.1.3	ptxqRegQMax	INTEGER	read
1.1.4	ptxqExpQSize	Gauge	read
1.1.5	ptxqExpQMax	INTEGER	read
1.1.6	ptxqQHpCount	Counter	read
1.1.7	ptxqQExpCount	Counter	read
1.1.8	ptxqQRegCount	Counter	read
1.1.9	ptxqQHpDiscards	Counter	read
1.1.10	ptxqQExpDiscards	Counter	read
1.1.11	ptxqQRegDiscards	Counter	read
1.1.12	ptxqMultiQSize	Gauge	read
1.1.13	ptxqMultiQMax	INTEGER	read
1.1.14	ptxqMultiQDiscards	Counter	read

Table A-14
pmsg GROUP

Device Pending Message Service
norand.manage.norandNet.nInterfaces.pmsg.x
(1.3.6.1.4.1.469.1000.2.2.6.x)

OID	**Object Name**	**Object Type**	**Access**
1.1.1	pmsgPort	INTEGER	read
1.1.2	pmsgPendRecCurrent	Gauge	read
1.1.3	pmsgPendRecMax	INTEGER	read
1.1.4	pmsgPendMsgCurrent	Gauge	read
1.1.5	pmsgPendMsgMax	INTEGER	read
1.1.6	pmsgPendMsgTotal	Counter	read
1.1.7	pmsgPendMsgDiscards	Counter	read
1.1.8	pmsgPendRecOverflowErrors	Counter	read
1.1.9	pmsgPendMsgOverflowErrors	Counter	read
1.1.10	pmsgPendAgedRecCount	Counter	read
1.1.11	pmsgPendAgedMsgCount	Counter	read

SNMP Version 1 Configuration Group

This group contains objects that configure the version 1 Simple Network Management Protocol (SNMP) agent.

" **NOTE:** *The MIB definition for each group starts on the page given below.*

" community	Community Table (page A-59)
" trapTarget	Trap Target Table (page A-61)

Table A-15
community TABLE

Device SNMP v1 Configurations
norand.manage.norandNet.nSNMP.v1Config.x
(1.3.6.1.4.1.469.1000.2.11.1.x)

OID	Object Name	Object Type	Access
2.1.1	communityIndex	INTEGER	read
2.1.2	communityStatus	INTEGER	write
2.1.3	communityName	DisplayString	write
2.1.4	communityPrivileges	INTEGER	write

Table A-16
trapTarget TABLE

Device SNMP v1 Configurations
norand.manage.norandNet.nSNMP.v1Config.x
(1.3.6.1.4.1.469.1000.2.11.1.x)

OID	Object Name	Object Type	Access
3.1.1	trapTargetIndex	INTEGER	read
3.1.2	trapTargetStatus	INTEGER	write
3.1.3	trapTargetName	DisplayString	write
3.1.4	trapTargetIpAddress	IpAddress	write

Bridging Parameters

The following groups contain objects relating to the wireless transparent bridging operation.

" **NOTE:** *The MIB definition for each group starts on the page given below.*

" rt	Route Table (page A-63)
" brg	Bridge Table (page A-66)
" addr	Address Table (page A-68)
" brgState	Bridge State Information (page A-70)
" bridgeStats	Bridge Statistics (page A-74)

Table A-17
rt GROUP

Device Route Table
norand.manage.norandNet.nBridge.rt.x
(1.3.6.1.4.1.469.1000.2.17.2.x)

OID	Object Name	Object Type	Access
2.1.1	rtDestination	PhysAddress	read
2.1.2	rtPort	INTEGER	read
2.1.3	rtAge	INTEGER	read
2.1.4	rtNodeId	INTEGER	read
2.1.5	rtAttachId	INTEGER	read
2.1.6	rtAttachTime	TimeTicks	read
2.1.7	rtApEaddr	PhysAddress	read
2.1.8	rtHopAddrLen	INTEGER	read
2.1.9	rtHopAddr16	INTEGER	read

Table A-17 (Continued)
rt GROUP

Device Route Table
norand.manage.norandNet.nBridge.rt.x
(1.3.6.1.4.1.469.1000.2.17.2.x)

OID	Object Name	Object Type	Access
2.1.10	rtHopEaddr	PhysAddress	read
2.1.11	rtIsBound	INTEGER	read
2.1.12	rtIsRemote	INTEGER	read
2.1.13	rtIsChild	INTEGER	read
2.1.14	rtIsAp	INTEGER	read
2.1.15	rtIsDistributed	INTEGER	read
2.1.16	rtIsRemoteLan	INTEGER	read
2.1.17	rtNS	INTEGER	read
2.1.18	rtNR	INTEGER	read

Table A-18
brg GROUP

Device Bridge Table
norand.manage.norandNet.nBridge.brg.x
(1.3.6.1.4.1.469.1000.2.17.3.x)

OID	Object Name	Object Type	Access
2.1.1	brgDestination	PhysAddress	read
2.1.2	brgPort	INTEGER	read
2.1.3	brgAge	INTEGER	read
2.1.4	brgType	INTEGER	read
2.1.5	brgIsPermanent	INTEGER	read
2.1.6	brgTimestamp	TimeTicks	read

Table A-19

addr GROUP

Address Table
norand.manage.norandNet.nBridge.addr.x
(1.3.6.1.4.1.469.1000.2.17.4.x)

OID	Object Name	Object Type	Access
2.1.1	addrDestination	PhysAddress	read
2.1.2	addrAge	INTEGER	read
2.1.3	addrNodeId	INTEGER	read
2.1.4	addrAlias	DisplayString	read
2.1.5	addrDeviceId	INTEGER	read
2.1.6	addrIpAddress	IPAddress	read

Table A-20

brgState GROUP

Bridge State Information
norand.manage.norandNet.nBridge.brgState.x
(1.3.6.1.4.1.469.1000.2.17.6.x)

OID	Object Name	Object Type	Access
3	bsAddress	PhysAddress	read
4	bsLanId	INTEGER	read
5	bsCostToRoot	INTEGER	read
6	bsIsRoot	INTEGER	read
7	bsIsAttached	INTEGER	read
8	bsAttachId	INTEGER	read
9	bsMyRootPriority	INTEGER	read
10	bsRootPort	INTEGER	read

Table A-20 (Continued)
brgState GROUP

Bridge State Information
norand.manage.norandNet.nBridge.brgState.x
(1.3.6.1.4.1.469.1000.2.17.6.x)

OID	Object Name	Object Type	Access
11	bsDesignatedRootAddress	PhysAddress	read
12	bsDesignatedRootPriority	INTEGER	read
13	bsDesignatedRootSequence	INTEGER	read
14	bsParentAddress	PhysAddress	read
15	bsPortCount	INTEGER	read
16	bsNodeId	INTEGER	read
17	bsRootChangedCount	Counter	read
18	bsRootCount	Counter	read
19	bsAttachCount	Counter	read
20	bsDetachReason	INTEGER	read
21	bsNetworkTime	TimeTicks	read
22	bsUniFloodLevel	INTEGER	read
23	bsMultiFloodLevel	INTEGER	read
24	bsIsPrimaryBridge	INTEGER	read
25	bsIsSecondaryBridge	INTEGER	read
26	bsUniFilterExpr	INTEGER	read
27	bsMultiFilterExpr	INTEGER	read

Table A-21
bridgeStats GROUP

Bridge Statistics
norand.manage.norandNet.nBridge.bridgeStats.x
(1.3.6.1.4.1.469.1000.2.17.7.x)

OID	Object Name	Object Type	Access
3	bstcRouteCount	Gauge	read
4	bstcChildCount	Gauge	read
5	bstcChildApCount	Gauge	read
6	bstcRemoteCount	Gauge	read
7	bstcPrimaryCount	Gauge	read
8	bstcInboundCount	Gauge	read
9	bstcSecondaryCount	Gauge	read
10	bstcRemoteLanCount	Gauge	read
11	bstcRouteGetErrors	Counter	read
12	bstcEntryGetErrors	Counter	read
13	bstcRmtLanGetErrors	Counter	read
14	bstcRouteSeqErrors	Counter	read
15	bstcDeleteSeqErrors	Counter	read
16	bstcEntrySeqErrors	Counter	read
17	bstcInvalidUpdateErrors	Counter	read

Transport Groups

The following groups contain objects that relate active transport layers information.

wst	WTP Status Table (page A-76)
hlit	High Level Interface Table (page A-82)
wcbst	WTP Control Block Status Table (page A-83)

Table A-22
wst GROUP

WTP Status Table
norand.manage.norandNet.nTransport.wst.x
(1.3.6.1.4.1.469.1000.2.102.2.x)

OID	Object Name	Object Type	Access
1	wstReset	INTEGER	read
2	wstLastReset	TimeTicks	read
3	wstClockTicks	TimeTicks	read
4	wstInternalErrorCount	Counter	read
5	wstSessionsLost	Counter	read
6	wstFrmrCount	Counter	read
7	wstDataBytesXmit	Counter	read
8	wstDataBytesRecv	Counter	read
9	wstIframesXmit	Counter	read
10	wstIframesRecv	Counter	read
11	wstIframesRetransmitted	Counter	read
12	wstCheckSumErrors	Counter	read
13	wstHIDiscardCount	Counter	read
14	wstState	INTEGER	read
15	wstBrgBusy	INTEGER	read
16	wstT1Timeouts	Counter	read
17	wstT1Bucket1	Counter	read
18	wstT1Bucket2	Counter	read
19	wstT1Bucket3	Counter	read
20	wstT1Bucket4	Counter	read
21	wstSABMCount	Counter	read

Table A-22 (Continued)
wst GROUP

WTP Status Table
norand.manage.norandNet.nTransport.wst.x
(1.3.6.1.4.1.469.1000.2.102.2.x)

OID	Object Name	Object Type	Access
22	wstDiscCount	Counter	read
23	wstDMCount	Counter	read
24	wstRNRCount	Counter	read
25	wstRejectCount	Counter	read
26	wstSessionsReset	Counter	read
27	wstBrgSrvTime	TimeTicks	read
28	wstBrgSrvCount	Counter	read
29	wstBrgSrvThreshold	TimeTicks	read
30	wstBrgSrvThreshCount	Counter	read
31	wstBrgSrvLongest	TimeTicks	read
32	wstBrgTxErrors	Counter	read
33	wstFatalBrgErrors	Counter	read

Table A-23
hlit GROUP

High Level Interface Table
norand.manage.norandNet.nTransport.hlit.x
(1.3.6.1.4.1.469.1000.2.102.3.x)

OID	Object Name	Object Type	Access
1	hlitReset	INTEGER	write
2	hlitLastReset	TimeTicks	read
3	hlitTime	TimeTicks	read
4	hlitCount	Counter	read
5	hlitThreshold	TimeTicks	write
6	hlitThreshCount	Counter	read
7	hlitLongest	Counter	read

Table A-24
wcbst GROUP

WTP Control Block Status Table
norand.manage.norandNet.nTransport.wcbst.x
(1.3.6.1.4.1.469.1000.2.102.4.x)

OID	Object Name	Object Type	Access
1	wcbstReset	INTEGER	write
2	wcbstLastReset	TimeTicks	read
3.1.1	wcbstIndex	INTEGER	read
3.1.2	wcbstInUse	INTEGER	read
3.1.3	wcbstH1State	INTEGER	read
3.1.4	wcbstDsap	INTEGER	read
3.1.5	wcbstSsap	INTEGER	read
3.1.6	wcbstNet16ADDR	INTEGER	read
3.1.7	wcbstT1Value	INTEGER	read
3.1.8	wcbstT1Average	INTEGER	read
3.1.9	wcbstT1Deviation	INTEGER	read
3.1.10	wcbstT1Timeouts	Counter	write
3.1.11	wcbstT2Timeouts	Counter	write
3.1.12	wcbstTiTimeouts	Counter	write
3.1.13	wcbstFramesRecv	Counter	write
3.1.14	wcbstIframesRecv	Counter	write
3.1.15	wcbstIframesDiscarded	Counter	write
3.1.16	wcbstDataBytesRecv	Counter	write
3.1.17	wcbstFramesXmit	Counter	write
3.1.18	wcbstIframesXmit	Counter	write
3.1.19	wcbstDataBytesXmit	Counter	write
3.1.20	wcbstIframesRetransmitted	Counter	write
3.1.21	wcbstState	INTEGER	read

Application Layer Groups

These groups relate resident applications information:

- Gateway application:
  - gs — Gateway Statistics (page A-88)
  - rcbs — Radio Terminal Control Block Statistics (page A-94)
  - hd — Host Delay Statistics (page A-100)
  - td — Terminal Transaction Delay Statistics (page A-102)
- Asynchronous host:
  - ahost — Asynchronous Host application (page A-104)
  - tgate — Gateway Connection Table (page A-107; *applies only to the 6910 Telnet Gateway/Access Point)*

Table A-25
gs GROUP

Gateway Statistics
norand.manage.norandNet.nApplication.gwSession.gs.x
(1.3.6.1.4.1.469.1000.2.104.1.1.x)

OID	Object Name	Object Type	Access
1	gsReset	INTEGER	write
2	gsLastReset	TimeTicks	read
3	gsClockTicks	TimeTicks	read
4	gsNetworkErrors	Counter	read
5	gsHostDataCount	Counter	read
6	gsHostHaltCount	Counter	read
7	gsHostInvalidCount	Counter	read
8	gsHostDiscardCount	Counter	read
9	gsConnectCount	Counter	read
10	gsCloseCount	Counter	read
11	gsBlocksXmit	Counter	read

Table A-25 (Continued)
gs GROUP

Gateway Statistics
norand.manage.norandNet.nApplication.gwSession.gs.x
(1.3.6.1.4.1.469.1000.2.104.1.1.x)

OID	Object Name	Object Type	Access
12	gsDataXmit	Counter	read
13	gsBlocksRecv	Counter	read
14	gsDataRecv	Counter	read
15	gsTermResetCount	Counter	read
16	gsTermContinueCount	Counter	read
17	gsTermInvalidCount	Counter	read
18	gsConnectionErrors	Counter	read
19	gsConnectionResets	Counter	read
20	gsRetransmissions	Counter	read
21	gsRecvSeqErrors	Counter	read
22	gsXmitErrors	Counter	read
23	gsTermDiscardCount	Counter	read
24	gsHostDelayMax	INTEGER	write
25	gsHostDelayTotal	Counter	read
26	gsHostTransactions	Counter	read
27	gsRecvErrors	Counter	read
28	gsTermDelayMax	INTEGER	write
29	gsTermDelayThreshold	INTEGER	write
30	gsTermDelayTotal	Counter	read
31	gsTermTransactions	Counter	read
32	gsTermThresholdCount	Counter	read
33	gsTermMaxCount	Counter	read
34	gsHostMaxCount	Counter	read
35	gsTermDelayTraceOn	INTEGER	write
36	gsHostActive	INTEGER	read

Table A-26
rcbs GROUP

Gateway (Terminal Emulation) Session Table
(each record is a terminal session)
norand.manage.norandNet.nApplication.gwSession.rcbs.x
(1.3.6.1.4.1.469.1000.2.104.1.2.x)

OID	Object Name	Object Type	Access
4.1.1	rcbsIndex	INTEGER	read
4.1.2	rcbsTerminal	INTEGER	read
4.1.3	rcbsType	INTEGER	read
4.1.4	rcbsState	INTEGER	read
4.1.5	rcbsLLCIndex	INTEGER	read
4.1.6	rcbsHostDataCount	Counter	read
4.1.7	rcbsBlocksXmit	Counter	read
4.1.8	rcbsDataXmit	Counter	read
4.1.9	rcbsBlocksRecv	Counter	read
4.1.10	rcbsDataRecv	Counter	read
4.1.11	rcbsTermResetCount	Counter	read
4.1.12	rcbsTermContinueCount	Counter	read
4.1.13	rcbsCloseCount	Counter	read
4.1.14	rcbsHostTransactions	Counter	read
4.1.15	rcbsHostDelayTotal	Counter	read
4.1.16	rcbsHostDelayLast	INTEGER	read
4.1.17	rcbsTermResetTime	INTEGER	read
4.1.18	rcbsTermInactTime	INTEGER	read
4.1.19	rcbsHostInactTime	INTEGER	read
4.1.20	rcbsConnectionErrors	Counter	read
4.1.21	rcbsNetAddress	PhysAddress	read
4.1.22	rcbsSeqErrors	Counter	read
4.1.23	rcbsTermTransactions	Counter	read
4.1.24	rcbsTermDelayTotal	Counter	read
4.1.25	rcbsTermDelayLast	INTEGER	read
4.1.26	rcbsTermDelayLongest	INTEGER	read
4.1.27	rcbsTermThresholdCount	Counter	read

Table A-26 (Continued)

rcbs GROUP

Gateway (Terminal Emulation) Session Table
(each record is a terminal session)
norand.manage.norandNet.nApplication.gwSession.rcbs.x
(1.3.6.1.4.1.469.1000.2.104.1.2.x)

OID	Object Name	Object Type	Access
4.1.28	rcbsTermMaxCount	Counter	read
4.1.29	rcbsTermDelayTraceOn	INTEGER	write
4.1.30	rcbsHostAlias	DisplayString	read

Table A-27

hd GROUP

Host Delay Statistics
norand.manage.norandNet.nApplication.gwSession.hd.x
(1.3.6.1.4.1.469.1000.2.104.1.3.x)

OID	Object Name	Object Type	Access
4.1.1	hdHostDelay	INTEGER	write
4.1.2	hdCount	Counter	read
4.1.3	hdBucket	INTEGER	read

Table A-28

td GROUP

Terminal Transaction Delay Statistics
norand.manage.norandNet.nApplication.gwSession.td.x
(1.3.6.1.4.1.469.1000.2.104.1.4.x)

OID	Object Name	Object Type	Access
4.1.1	tdTrxnTime	INTEGER	write
4.1.2	tdCount	Counter	read
4.1.3	tdBucket	INTEGER	read

Table A-29
ahost GROUP

Asynchronous Host
norand.manage.norandNet.nApplication.ahost.x
(1.3.6.1.4.1.469.1000.2.104.2.x)

OID	Object Name	Object Type	Access
1	ahstReset	INTEGER	write
2	ahstLastReset	TimeTicks	read
3	ahstFramesOut	Counter	read
4	ahstFramesIn	Counter	read
5	ahstIdleDetected	Counter	read
6	ahstNoErrors	Counter	read
7	ahstSyntaxErrors	Counter	read
8	ahstLengthErrors	Counter	read
9	ahstDisabledErrors	Counter	read
10	ahstGapErrors	Counter	read
11	ahstCheckErrors	Counter	read
12	ahstNoBuffersErrors	Counter	read
13	ahstSequenceErrors	Counter	read
15	ahstParityErrors	Counter	read
16	ahstFrameErrors	Counter	read
17	ahstNoiseErrors	Counter	read
18	ahstBreakErrors	Counter	read

Table A-30
tgate GROUP

(Applies only to the 6910 Telnet Gateway/Access Point)

Gateway Connection Table
norand.manage.norandNet.nApplication.tgate.x
(1.3.6.1.4.1.469.1000.2.104.3.x)

OID	Object Name	Object Type	Access
3.1.1	tgateIndex	INTEGER	read
3.1.2	tgateTerminal	INTEGER	read
3.1.3	tgateHostIpAddress	IpAddress	read
3.1.4	tgateHostPort	INTEGER	read
3.1.5	tgateGatePort	INTEGER	read
3.1.6	tgateEvent	INTEGER	read
3.1.7	tgateEventAge	TimeTicks	read
3.1.8	tgateIdleTime	TimeTicks	read
3.1.9	tgateOfflineTime	TimeTicks	read

Control Groups

Objects in the following groups exert control over the 6710 Access Point. Present functions include rebooting and scheduling software downloads.

" **NOTE:** *The MIB definition for each group starts on the page given below.*

" powerUp	Power Up Objects (page A-110)
" softwareDownLoad	Software Download (page A-110)

Table A-31
powerUp GROUP

Device Power Up Objects
norand.manage.norandNet.nControl.powerUp.x
(1.3.6.1.4.1.469.1000.2.105.1.x)

OID	Object Name	Object Type	Access
1	pwrPowerUpCount	Counter	read
2	pwrNextPowerUpTime	TimeTicks	write

Table A-32
softwareDownLoad GROUP

Device Software Download
norand.manage.norandNet.nControl.softwareDownLoad.x
(1.3.6.1.4.1.469.1000.2.105.2.x)

OID	Object Name	Object Type	Access
1	sdStartTime	TimeTicks	write
2	sdServerIpAddress	IpAddress	write
3	sdScriptFilename	DisplayString	write
4	sdStatus	INTEGER	read
5	sdErrorString	DisplayString	read
6	sdCheckPoint	INTEGER	write
7	sdSetActivePointers	INTEGER	write
8	sdTerminate	INTEGER	write

MIB Definitions

Following are the MIB definitions for the 6910 Integrated Gateway/Access Point and 6910 Telnet Gateway/Access Point.

```
 -- *****************************************************************
 -- *
 -- *                        6910IMIB.MIB Version 1.32
 -- *
 -- *****************************************************************
 -- *****************************************************************
 -- *
 -- *                        6910TMIB.MIB Version 1.32
 -- *
 -- *****************************************************************
OWL DEFINITIONS ::= BEGIN
      IMPORTS
          enterprises, IpAddress, Counter, Gauge, TimeTicks
              FROM RFC1155-SMI
          PhysAddress, DisplayString
              FROM RFC1213-MIB
          OBJECT-TYPE
              FROM RFC-1212;
      --  This MIB module uses the extended OBJECT-TYPE macro as
      --  defined in RFC-1212;
norand                          OBJECT IDENTIFIER ::= { enterprises 469 }
    manage                      OBJECT IDENTIFIER ::= { norand 1000 }
        products                OBJECT IDENTIFIER ::= { manage 1 }
            ap6710              OBJECT IDENTIFIER ::= { products 1 }
            gw4030              OBJECT IDENTIFIER ::= { products 2 }
            wnas                OBJECT IDENTIFIER ::= { products 3 }
            ts6950              OBJECT IDENTIFIER ::= { products 4 }
            gwap6910            OBJECT IDENTIFIER ::= { products 5 }
            uap2100             OBJECT IDENTIFIER ::= { products 6 }
            msd6710             OBJECT IDENTIFIER ::= { products 7 }
        norandNET               OBJECT IDENTIFIER ::= { manage 2 }
            nSystem             OBJECT IDENTIFIER ::= { norandNET 1 }
                 hw              OBJECT IDENTIFIER ::= { nSystem 1 }
                    -- The Hardware Parameters Group
                    hwPartNo      OBJECT-TYPE
                     SYNTAX INTEGER (0..2147483647)
                     ACCESS read-only
                     STATUS mandatory
                     DESCRIPTION
                     ”The Norand part number of the hardware device.”
                     ::= { hw 1 }
                    hwDescription OBJECT-TYPE
                     SYNTAX DisplayString (SIZE (0..40))
                     ACCESS read-only
                     STATUS mandatory
                     DESCRIPTION
                     ”The description of the hardware device.”
                     ::= { hw 2 }
```

```
     hwRevision    OBJECT-TYPE
      SYNTAX INTEGER (0..2147483647)
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      "The revision level of the hardware device."
      ::= { hw 3 }
     hwSerialNo    OBJECT-TYPE
      SYNTAX INTEGER (0..2147483647)
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      "The serial number of the hardware device."
      ::= { hw 4 }
     hwID          OBJECT-TYPE
      SYNTAX INTEGER (0..2147483647)
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      "The device identifier of the hardware device. Values = 3250,
      4000, 4020, 4030, 4033, 3240, 1000, 1100, 1700, 5940, 4650,
      100 (ACE process), 200 (DOSNMS), 300 (Norand Proxy Agent),
      6710 (Access Point)."
      ::= { hw 5 }
file               OBJECT IDENTIFIER ::= { nSystem 3 }
     fsinfo         OBJECT IDENTIFIER ::= { file 1 }
          -- The FileSystem Information Table
          fsEnabled     OBJECT-TYPE
            SYNTAX INTEGER { true(1), false(2) }
            ACCESS read-only
            STATUS mandatory
            DESCRIPTION
            "TRUE, if the file system is enabled"
            ::= { fsinfo 1 }
          fsMaxSectors  OBJECT-TYPE
            SYNTAX INTEGER
            ACCESS read-only
            STATUS mandatory
            DESCRIPTION
            "The number of physical sectors.  A file segment
            consists of one or more adjacent physical sectors."
            ::= { fsinfo 2 }
          fsSectorSize  OBJECT-TYPE
            SYNTAX INTEGER
            ACCESS read-only
            STATUS mandatory
            DESCRIPTION
            "The size of a physical sector in bytes."
            ::= { fsinfo 3 }
```

```
    fsNumSegments OBJECT-TYPE
      SYNTAX INTEGER
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      ”The number of logical file segments
      (0-MAX_SECTORS)”
      ::= { fsinfo 4 }

    fsNumFiles    OBJECT-TYPE
      SYNTAX Gauge
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      ”The number of files (0-25)”
      ::= { fsinfo 5 }

    fsBootSegment OBJECT-TYPE
      SYNTAX INTEGER
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      ”The index of the current boot segment.  If the index is
      non-zero and the first file in the associated segment is
      executable, then control is passed to that file during
      the power-up sequence.”
      ::= { fsinfo 6 }

    fsDataSegment OBJECT-TYPE
      SYNTAX INTEGER
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      ”The index of the active data segment.  Files stored in
      this segment will be accessible to an executing
      application.”
      ::= { fsinfo 7 }

segment        OBJECT IDENTIFIER ::= { file 2 }

-- The File Segment Table

    -- Table Definition

    segTable       OBJECT-TYPE
     SYNTAX SEQUENCE OF SEGEntry
     ACCESS not-accessible
     STATUS mandatory
     DESCRIPTION
     ””
     ::= { segment 2 }

    -- Row Definition
```

```
segEntry       OBJECT-TYPE
 SYNTAX SEGEntry
 ACCESS not-accessible
 STATUS mandatory
 INDEX { segID }
 ::= { segTable 1 }
-- Columnar Object Definitions
SEGEntry ::=
 SEQUENCE {
   segID           INTEGER,
   segFirstSector  INTEGER,
   segLastSector   INTEGER,
   segStatus       INTEGER,
   segSize         INTEGER,
   segFree         INTEGER
}
segID          OBJECT-TYPE
  SYNTAX INTEGER
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  "The segment ID (1 - (NUM_SEGMENTS+1)).  A non-zero
  number which uniquely identifies a segment."
  ::= { segEntry 1 }
segFirstSector OBJECT-TYPE
  SYNTAX INTEGER
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  "The first physical sector in the segment
  (1 - (MAX_SECTORS + 1))"
  ::= { segEntry 2 }
segLastSector  OBJECT-TYPE
  SYNTAX INTEGER
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  "The last physical sector in the segment
  (FIRST_SECTOR - (MAX_SECTORS + 1))"
  ::= { segEntry 3 }
segStatus      OBJECT-TYPE
  SYNTAX INTEGER  {   valid(1),
                    invalid(2) }
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  "The segment status:
     valid   = 1,
     invalid = 2 "
  ::= { segEntry 4 }
```

```
    segSize        OBJECT-TYPE
      SYNTAX INTEGER
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      ”The segment size in bytes”
      ::= { segEntry 5 }

    segFree        OBJECT-TYPE
      SYNTAX INTEGER
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      ”The number of available bytes in the segment which are
      not currently allocated to a file.”
      ::= { segEntry 6 }

dir            OBJECT IDENTIFIER ::= { file 3 }

-- The File Directory Table

    -- Table Definition

    dirTable   OBJECT-TYPE
     SYNTAX SEQUENCE OF DIREntry
     ACCESS not-accessible
     STATUS mandatory
     DESCRIPTION
     ”The FileSystem Directory”
     ::= { dir 2 }

    -- Row Definition

    dirEntry   OBJECT-TYPE
     SYNTAX DIREntry
     ACCESS not-accessible
     STATUS mandatory
     INDEX { dirIndex }
     ::= { dirTable 1 }

    -- Columnar Object Definitions

    DIREntry ::=
     SEQUENCE {
       dirIndex   INTEGER,
       dirName    DisplayString,
       dirSegment INTEGER,
       dirType    INTEGER,
       dirSize    INTEGER,
       dirDate    DisplayString,
       dirTime    DisplayString,
       dirVersion DisplayString
    }
```

```
dirIndex   OBJECT-TYPE
  SYNTAX INTEGER
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Directory Index”
  ::= { dirEntry 1 }

dirName    OBJECT-TYPE
  SYNTAX DisplayString (SIZE (0..14))
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”File name”
  ::= { dirEntry 2 }

dirSegment OBJECT-TYPE
  SYNTAX INTEGER
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”File segment (1 - (NUM_SEGMENTS + 1)).
  The segment ID which identifies the segment containing
  the file.”
  ::= { dirEntry 3 }

dirType    OBJECT-TYPE
  SYNTAX INTEGER { executable(1),
                        data(2),
                     invalid(3) }
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”File type:
     executable = 1,
     data       = 2,
     invalid    = 3 ”
  ::= { dirEntry 4 }

dirSize    OBJECT-TYPE
  SYNTAX INTEGER
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The file size in bytes”
  ::= { dirEntry 5 }

dirDate    OBJECT-TYPE
  SYNTAX DisplayString (SIZE (0..12))
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The file date in MM-DD-YYYY display format.”
  ::= { dirEntry 6 }
```

```
        dirTime    OBJECT-TYPE
          SYNTAX DisplayString (SIZE (0..10))
          ACCESS read-only
          STATUS mandatory
          DESCRIPTION
          "The file time in HH:MM:SS display format."
          ::= { dirEntry 7 }

        dirVersion OBJECT-TYPE
          SYNTAX DisplayString (SIZE (0..8))
          ACCESS read-only
          STATUS mandatory
          DESCRIPTION
          "The file version in v99.99 display format."
          ::= { dirEntry 8 }

sysErrors          OBJECT IDENTIFIER ::= { nSystem 4 }

    criticalErrors OBJECT IDENTIFIER ::= { sysErrors 1 }

        ceEnabled        OBJECT-TYPE
          SYNTAX INTEGER { true(1), false(2) }
          ACCESS read-only
          STATUS mandatory
          DESCRIPTION
          "A value of true(1) signifies that the critical error
          log was successfully initialized as part of the power-up
          sequence.  Any errors in that initialization process
          result in a value of false(2)."
          ::= { criticalErrors 1 }

        ceOverflow       OBJECT-TYPE
          SYNTAX INTEGER
          ACCESS read-only
          STATUS mandatory
          DESCRIPTION
          "Overflow error code.  If the overflow code is non-zero,
          it indicates that the log has overflowed and the
          overflow code contains the last displaced value."
          ::= { criticalErrors 2 }

        ceReset          OBJECT-TYPE
          SYNTAX INTEGER { true(1), false(2) }
          ACCESS read-write
          STATUS mandatory
          DESCRIPTION
          "A user can reset the critical error log by
          setting ceReset to true(1).  Valid values are
          true(1) or false(2)."
          ::= { criticalErrors 3 }
```

```
            ceLogTable      OBJECT-TYPE
              SYNTAX SEQUENCE OF CELogEntry
              ACCESS not-accessible
              STATUS mandatory
              DESCRIPTION
              ”Critical Error Log Table”
              ::= { criticalErrors 4 }

            ceLogEntry      OBJECT-TYPE
              SYNTAX CELogEntry
              ACCESS not-accessible
              STATUS mandatory
              INDEX { ceLogErrorCode }
              ::= { ceLogTable 1 }

            CELogEntry ::=
              SEQUENCE {
                ceLogErrorCode  INTEGER,
                ceLogErrorCount Counter
              }

            ceLogErrorCode  OBJECT-TYPE
              SYNTAX INTEGER
              ACCESS read-only
              STATUS mandatory
              DESCRIPTION
              ”Critical error code.  A 16-bit value which uniquely
              indentifies a system software error.  The error codes
              are intended for internal Norand use.”
              ::= { ceLogEntry 1 }

            ceLogErrorCount OBJECT-TYPE
              SYNTAX Counter
              ACCESS read-only
              STATUS mandatory
              DESCRIPTION
              ”Error count for the associated ceLogErrorCode”
              ::= { ceLogEntry 2 }

nInterfaces            OBJECT IDENTIFIER ::= { norandNET 2 }

   nifx               OBJECT IDENTIFIER ::= { nInterfaces 2 }

-- The Norand Extended Interfaces Table

        nifxTable              OBJECT-TYPE
         SYNTAX SEQUENCE OF NIFXEntry
         ACCESS not-accessible
         STATUS mandatory
         DESCRIPTION
         ”Norand Extended Interface Table”
         ::= { nifx 4 }
```

```
nifxEntry              OBJECT-TYPE
 SYNTAX NIFXEntry
 ACCESS not-accessible
 STATUS mandatory
 INDEX { nifxIndex }
 ::= { nifxTable 1 }

NIFXEntry ::=
  SEQUENCE {
    nifxIndex                  INTEGER,
    nifxType                   INTEGER,
    nifxInDisabledDiscards     Counter,
    nifxInOverruns             Counter,
    nifxInHWOverruns           Counter,
    nifxInUcastDPkts           Counter,
    nifxInNUcastDPkts          Counter,
    nifxInLenErrors            Counter,
    nifxExcessiveDeferrals     Counter,
    nifxInNetIDDiscards        Counter,
    nifxInFragDiscards         Counter,
    nifxInUFilterDiscards      Counter,
    nifxInNUFilterDiscards     Counter,
    nifxInQFullDiscards        Counter
}

nifxIndex              OBJECT-TYPE
 SYNTAX INTEGER
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 "Interface index"
 ::={ nifxEntry 1 }

nifxType               OBJECT-TYPE
 SYNTAX INTEGER {         ether(4),
                         bb485(33),
                         owlIP(66),
                    proxim24(132),
                        nor24(195),
                  falcon902(197),
                          uhf(198)
                }
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 "Norand Interface Type"
 ::={ nifxEntry 2 }
```

```
nifxInDisabledDiscards OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”The number of received unicast frames which do not require
 forwarding.  Unicast ethernet frames are discarded if
 ether-to-radio flooding is disabled and the destination is
 unknown; otherwise, unicast frames are discarded if the
 bridge has learned that the destination port is the same as
 the source port”
 ::={ nifxEntry 3 }

nifxInOverruns         OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”The number of received frames discarded because the frame
 could not be queued for the MAC-D task”
 ::={ nifxEntry 4 }

nifxInHWOverruns       OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”The number of received frames discarded due to hardware
 overruns.”
 ::={ nifxEntry 5 }

nifxInUcastDPkts       OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”The number of received unicast frames successfully delivered
 to the MAC-D task”
 ::={ nifxEntry 6 }

nifxInNUcastDPkts      OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”The number of received multicast frames successfully
 delivered to the MAC-D task”
 ::={ nifxEntry 7 }
```

```
nifxInLenErrors        OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”The number received frames with length errors”
 ::={ nifxEntry 8 }

nifxExcessiveDeferrals OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”The number of aborted transmissions due to excessive
 deferrals”
 ::={ nifxEntry 9 }

nifxInNetIDDiscards    OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”The number of received frames discarded because the LAN ID
 did not match”
 ::={ nifxEntry 10 }

nifxInFragDiscards     OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”The number of received frame fragments discarded because a
 fragmented frame could not be re-assembled”
 ::={ nifxEntry 11 }

nifxInUFilterDiscards  OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”The number of enabled received unicast frames discarded due
 to a unicast filter expression”
 ::={ nifxEntry 12 }

nifxInNUFilterDiscards OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”The number of enabled received multicast frames discarded
 due to a multicast filter expression”
 ::={ nifxEntry 13 }
```

```
     nifxInQFullDiscards    OBJECT-TYPE
      SYNTAX Counter
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      "The number of received frames discarded because the frame
      could not be queued for the MAC-R task"
      ::={ nifxEntry 14 }

portState           OBJECT IDENTIFIER ::= { nInterfaces 3 }

  -- The Port State Table

     psTable                  OBJECT-TYPE
      SYNTAX SEQUENCE OF PSEntry
      ACCESS not-accessible
      STATUS mandatory
      DESCRIPTION
      "MAC-R port state variables"
      ::= { portState 4 }

     psEntry                  OBJECT-TYPE
      SYNTAX PSEntry
      ACCESS not-accessible
      STATUS mandatory
      INDEX { psPort }
      ::= { psTable 1 }

     PSEntry ::=
       SEQUENCE {
         psPort                     INTEGER,
         psIfIndex                  INTEGER,
         psAddress                  PhysAddress,
         psType                     INTEGER,
         psState                    INTEGER,
         psCost                     INTEGER,
         psHelloPeriod              INTEGER,
         psHelloCount               Counter,
         psMacdWindow               INTEGER,
         psMacdQSize                Gauge,
         psMacdTimeouts             Counter,
         psIsPrimary                INTEGER,
         psIsSecondary              INTEGER,
         psIsSecondaryCandidate     INTEGER,
         psSecondaryUniFlooding     INTEGER,
         psSecondaryMultiFlooding   INTEGER,
         psIsRadio                  INTEGER,
         psPendEnabled              INTEGER
     }
```

```
psPort                   OBJECT-TYPE
  SYNTAX INTEGER (1..4)
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”MAC-R port ID (1-4).  A number which uniquely identifies
  the port.”
  ::= { psEntry  1 }

psIfIndex                OBJECT-TYPE
  SYNTAX INTEGER
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”MAC-D interface index.  The index matches the interface
  index of the associated row in the mib-II interface table.”
  ::= { psEntry  2 }

psAddress                OBJECT-TYPE
  SYNTAX PhysAddress
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”802 address of the port”
  ::= { psEntry  3 }

psType                   OBJECT-TYPE
  SYNTAX INTEGER {        ether(4),
                         bb485(33),
                         owlIP(66),
                     proxim24(132),
                        nor24(195),
                    falcon902(197),
                          uhf(198)
                 }
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Norand port type:
    ether  = 4,
    Proxim = 196,
    Falcon = 197,
    UHF    = 198”
  ::= { psEntry  4 }
```

```
psState                 OBJECT-TYPE
  SYNTAX INTEGER { disabled(0),
                     idle(1),
                     open(2),
                  receive(3),
                 transmit(4) }
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  "Port state:
   disabled = 0,
   idle     = 1,
   open     = 2,
   receive  = 3,
   transmit = 4 "
  ::= { psEntry  5 }

psCost                  OBJECT-TYPE
  SYNTAX INTEGER
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  "Incremental path cost of the port.
  Default values:
   ether  = 20,
   Falcon = 100,
   UHF    = 255 "
  ::= { psEntry  6 }

psHelloPeriod           OBJECT-TYPE
  SYNTAX INTEGER
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  "Inter-HELLO time (.01 secs.)"
  ::= { psEntry  7 }

psHelloCount            OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  "HELLO transmit count"
  ::= { psEntry  8 }

psMacdWindow            OBJECT-TYPE
  SYNTAX INTEGER
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  "Max. number of active MAC-D transmit requests"
  ::= { psEntry  9 }
```

```
psMacdQSize             OBJECT-TYPE
  SYNTAX Gauge
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Current number of active MAC-D transmit requests”
  ::= { psEntry  10 }

psMacdTimeouts          OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”MAC-D transmit timeout errors”
  ::= { psEntry 11 }

psIsPrimary             OBJECT-TYPE
  SYNTAX INTEGER { true(1), false(2) }
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”TRUE, for primary bridge ports”
  ::= { psEntry 12 }

psIsSecondary           OBJECT-TYPE
  SYNTAX INTEGER { true(1), false(2) }
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”TRUE, for secondary bridge ports”
  ::= { psEntry 13 }

psIsSecondaryCandidate  OBJECT-TYPE
  SYNTAX INTEGER { true(1), false(2) }
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”TRUE, if secondary bridge port candidates”
  ::= { psEntry 14 }

psSecondaryUniFlooding  OBJECT-TYPE
  SYNTAX INTEGER { true(1), false(2) }
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”TRUE, for secondary bridge ports which require unicast
  flooding”
  ::= { psEntry 15 }
```

```
     psSecondaryMultiFlooding OBJECT-TYPE
       SYNTAX INTEGER { true(1), false(2) }
       ACCESS read-only
       STATUS mandatory
       DESCRIPTION
       ”TRUE, for secondary bridge ports which require multicast
       flooding”
       ::= { psEntry 16 }

     psIsRadio                OBJECT-TYPE
       SYNTAX INTEGER { true(1), false(2) }
       ACCESS read-only
       STATUS mandatory
       DESCRIPTION
       ”TRUE, for radio ports”
       ::= { psEntry 17 }

     psPendEnabled            OBJECT-TYPE
       SYNTAX INTEGER { true(1), false(2) }
       ACCESS read-only
       STATUS mandatory
       DESCRIPTION
       ”TRUE, if the port supports pending messages”
       ::= { psEntry 18 }

portStats          OBJECT IDENTIFIER ::= { nInterfaces 4 }

     pstcTable                OBJECT-TYPE
      SYNTAX SEQUENCE OF PSTCEntry
      ACCESS not-accessible
      STATUS mandatory
      DESCRIPTION
      ”MAC-R port state variables”
      ::= { portStats 4 }

     pstcEntry                OBJECT-TYPE
      SYNTAX PSTCEntry
      ACCESS not-accessible
      STATUS mandatory
      INDEX { pstcPort }
      ::= { pstcTable 1 }
```

```
PSTCEntry ::=
  SEQUENCE {
    pstcPort                  INTEGER,
    pstcInOWLPkts             Counter,
    pstcInUcastOWLDataPkts    Counter,
    pstcInNUcastOWLDataPkts   Counter,
    pstcInOWLErrors           Counter,
    pstcOutOWLPkts            Counter,
    pstcOutUcastOWLDataPkts   Counter,
    pstcOutNUcastOWLDataPkts  Counter,
    pstcOutOWLErrors          Counter,
    pstcParentLinkErrors      Counter,
    pstcAlertLinkErrors       Counter,
    pstcInUcastRelayPkts      Counter,
    pstcInNUcastRelayPkts     Counter,
    pstcOutUcastRelayPkts     Counter,
    pstcOutNUcastRelayPkts    Counter,
    pstcInUcastInbound        Counter,
    pstcInUcastOutbound       Counter,
    pstcInUcastSec            Counter,
    pstcInUcastFlood          Counter,
    pstcInUcastDiscards       Counter,
    pstcInNUcastDiscards      Counter,
    pstcInUcastToIFC          Counter,
    pstcInNUcastToIFC         Counter,
    pstcOutDelayDiscards      Counter
}

pstcPort                  OBJECT-TYPE
  SYNTAX INTEGER (1..4)
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”MAC-R port ID (1-4).  A number which uniquely identifies
  the port.”
  ::= { pstcEntry 1 }

pstcInOWLPkts             OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Total received OWL packets”
  ::= { pstcEntry 2 }

pstcInUcastOWLDataPkts    OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Received OWL unicast data packets”
  ::= { pstcEntry 3 }
```

```
pstcInNUcastOWLDataPkts  OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Received multicast OWL data packets”
  ::= { pstcEntry 4 }
pstcInOWLErrors          OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Received OWL packets with errors”
  ::= { pstcEntry 5 }
pstcOutOWLPkts           OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Total OWL packets sent”
  ::= { pstcEntry 6 }
pstcOutUcastOWLDataPkts  OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Total unicast OWL data packets sent”
  ::= { pstcEntry 7 }
pstcOutNUcastOWLDataPkts OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Total multicast OWL data packets sent”
  ::= { pstcEntry 8 }
pstcOutOWLErrors         OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”OWL packet send errors”
  ::= { pstcEntry 9 }
pstcParentLinkErrors     OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Root port send link errors”
  ::= { pstcEntry  10 }
```

```
pstcAlertLinkErrors      OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Link errors which generated an ALERT”
  ::= { pstcEntry 11 }
pstcInUcastRelayPkts     OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Received unicast relay packets”
  ::= { pstcEntry 12 }
pstcInNUcastRelayPkts    OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Received multicast relay packets”
  ::= { pstcEntry 13 }
pstcOutUcastRelayPkts    OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Total unicast relay packets sent”
  ::= { pstcEntry 14 }
pstcOutNUcastRelayPkts   OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Total multicast relay packets sent”
  ::= { pstcEntry 15 }
pstcInUcastInbound       OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Received unicast data packets routed inbound or
  relayed onto the distribution LAN”
  ::= { pstcEntry 16 }
pstcInUcastOutbound      OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Received unicast data packets routed outbound”
  ::= { pstcEntry 17 }
```

```
pstcInUcastSec            OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Received unicast data packets relayed to a secondary LAN”
  ::= { pstcEntry 18 }
pstcInUcastFlood          OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Received unicast data packets with an unknown destination”
  ::= { pstcEntry 19 }
pstcInUcastDiscards       OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Received unicast data packets discarded”
  ::= { pstcEntry 20 }
pstcInNUcastDiscards      OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Received multicast data packets discarded”
  ::= { pstcEntry 21 }
pstcInUcastToIFC          OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Received unicast packets passed to the data link interface”
  ::= { pstcEntry 22 }
pstcInNUcastToIFC         OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Received multicast packets passed to the data link
  interface”
  ::= { pstcEntry 23 }
pstcOutDelayDiscards      OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Send packets discarded due to excessive delay”
  ::= { pstcEntry 24 }
```

```
ptxq          OBJECT IDENTIFIER ::= { nInterfaces 5 }

    ptxqTable          OBJECT-TYPE
      SYNTAX SEQUENCE OF PTXQEntry
      ACCESS not-accessible
      STATUS mandatory
      DESCRIPTION
      ”The Port Transmit Queue Table”
      ::= { ptxq 1 }

    ptxqEntry          OBJECT-TYPE
      SYNTAX PTXQEntry
      ACCESS not-accessible
      STATUS mandatory
      INDEX { ptxqPort }
      ::= { ptxqTable 1 }

    PTXQEntry ::=
      SEQUENCE {
        ptxqPort           INTEGER,
        ptxqRegQSize       Gauge,
        ptxqRegQMax        INTEGER,
        ptxqExpQSize       Gauge,
        ptxqExpQMax        INTEGER,
        ptxqQHpCount       Counter,
        ptxqQRegCount      Counter,
        ptxqQExpCount      Counter,
        ptxqQHpDiscards    Counter,
        ptxqQRegDiscards   Counter,
        ptxqQExpDiscards   Counter,
        ptxqMultiQSize     Gauge,
        ptxqMultiQMax      INTEGER,
        ptxqMultiQDiscards Counter
      }

    ptxqPort           OBJECT-TYPE
      SYNTAX INTEGER
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      ”MAC-R port ID (1-4).
       A number which uniquely indentifies the port.”
      ::= { ptxqEntry 1 }

    ptxqRegQSize       OBJECT-TYPE
      SYNTAX Gauge
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      ”Current regular queue size (0-REG_Q_MAX).
      The number of regular priority packets which are currently
      queued for transmission on the port.”
      ::= { ptxqEntry 2 }
```

```
ptxqRegQMax        OBJECT-TYPE
  SYNTAX INTEGER
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The maximum number of regular priority packets which can be
  queued for transmission on the port.”
  ::= { ptxqEntry 3 }

ptxqExpQSize       OBJECT-TYPE
  SYNTAX Gauge
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Current expedited queue size (0..EXP_Q_MAX).  The number of
  expedited packets which are currently queued for
  transmission on the port.”
  ::= { ptxqEntry 4 }

ptxqExpQMax        OBJECT-TYPE
  SYNTAX INTEGER
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The maximum number of expedited packets which can be queued
  for transmission on the port.”
  ::= { ptxqEntry 5 }

ptxqQHpCount       OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The number of attempts to queue a high priority packet for
  transmission”
  ::= { ptxqEntry 6 }

ptxqQExpCount      OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The number of attempts to queue an expedited priority
  packet for transmission”
  ::= { ptxqEntry 7 }

ptxqQRegCount      OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The number of attempts to queue a regular priority packet
  for transmission”
  ::= { ptxqEntry 8 }
```

```
ptxqQHpDiscards    OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The number of failed attemts to queue a high priority
  packet”
  ::= { ptxqEntry 9 }

ptxqQExpDiscards   OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The number of failed attemts to queue an expedited priority
  packet”
  ::= { ptxqEntry 10 }

ptxqQRegDiscards   OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The number of failed attemts to queue a regular priority
  packet”
  ::= { ptxqEntry 11 }

ptxqMultiQSize     OBJECT-TYPE
  SYNTAX Gauge
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Current multicast queue size.  The number of multicast
  packets which are queued for transmission on the (radio)
  port.  Multicast packets are transmitted after HELLO packets
  on OWL radio ports.”
  ::= { ptxqEntry 12 }

ptxqMultiQMax      OBJECT-TYPE
  SYNTAX INTEGER
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The maximum number of multicast packets which will be
  queued for transmission on the (radio) port.”
  ::= { ptxqEntry 13 }

ptxqMultiQDiscards OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The number of failed attempts to queue a Multicast packet”
  ::= { ptxqEntry 14 }
```

```
pmsg          OBJECT IDENTIFIER ::= { nInterfaces 6 }

    pmsgTable                 OBJECT-TYPE
      SYNTAX SEQUENCE OF PmsgEntry
      ACCESS not-accessible
      STATUS mandatory
      DESCRIPTION
      "Pending Message Table"
      ::= { pmsg 1 }

    pmsgEntry                 OBJECT-TYPE
      SYNTAX PmsgEntry
      ACCESS not-accessible
      STATUS mandatory
      INDEX { pmsgPort }
      ::= { pmsgTable 1 }

    PmsgEntry ::=
      SEQUENCE {
        pmsgPort                    INTEGER,
        pmsgPendRecCurrent          Gauge,
        pmsgPendRecMax              INTEGER,
        pmsgPendMsgCurrent          Gauge,
        pmsgPendMsgMax              INTEGER,
        pmsgPendMsgTotal            Counter,
        pmsgPendMsgDiscards         Counter,
        pmsgPendRecOverflowErrors Counter,
        pmsgPendMsgOverflowErrors Counter,
        pmsgPendAgedRecCount        Counter,
        pmsgPendAgedMsgCount        Counter
      }

    pmsgPort                  OBJECT-TYPE
      SYNTAX INTEGER
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      "MAC-R port ID (1-4).
      A number which uniquely indentifies the port."
      ::= { pmsgEntry 1 }

    pmsgPendRecCurrent        OBJECT-TYPE
      SYNTAX Gauge
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      "Current terminal record count"
      ::= { pmsgEntry 2 }
```

```
pmsgPendRecMax            OBJECT-TYPE
  SYNTAX INTEGER
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Maximum terminal record count”
  ::= { pmsgEntry 3 }

pmsgPendMsgCurrent        OBJECT-TYPE
  SYNTAX Gauge
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Current pending message count”
  ::= { pmsgEntry 4 }

pmsgPendMsgMax            OBJECT-TYPE
  SYNTAX INTEGER
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Maximum pending message count”
  ::= { pmsgEntry 5 }

pmsgPendMsgTotal          OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Total pending message count”
  ::= { pmsgEntry 6 }

pmsgPendMsgDiscards       OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The number of pending messages in-queue which were
  discarded before they could be delivered because the
  terminal’s queue was full.”
  ::= { pmsgEntry 7 }

pmsgPendRecOverflowErrors OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The number of times that a terminal requested pending
  message services when no pending message records were
  available.”
  ::= { pmsgEntry 8 }
```

```
          pmsgPendMsgOverflowErrors OBJECT-TYPE
            SYNTAX Counter
            ACCESS read-only
            STATUS mandatory
            DESCRIPTION
            "The number of times when the maximum number of stored
            messages, per platform, was exceeded."
            ::= { pmsgEntry 9 }

          pmsgPendAgedRecCount      OBJECT-TYPE
            SYNTAX Counter
            ACCESS read-only
            STATUS mandatory
            DESCRIPTION
            "The number of terminal records discarded due to maximum age
            (12 minutes)"
            ::= { pmsgEntry 10 }

          pmsgPendAgedMsgCount      OBJECT-TYPE
            SYNTAX Counter
            ACCESS read-only
            STATUS mandatory
            DESCRIPTION
            "The number of pending messages which were discarded due to
            maximum age. (default = 5 seconds)"
            ::= { pmsgEntry 11 }

nSNMP                  OBJECT IDENTIFIER ::= { norandNET 11 }

    v1Config           OBJECT IDENTIFIER ::= { nSNMP 1 }

      --Norand Community table defines the accepted community
      --strings and their access privileges

       -- The Community Table

          communityTable        OBJECT-TYPE
            SYNTAX  SEQUENCE OF CommunityEntry
            ACCESS  not-accessible
            STATUS  mandatory
            DESCRIPTION
            "The community table is used to define communities and their
            access privileges.  Norand's implementation of the community
            table has some special considerations:
```

```
     1) GETs and SETs to the community table can only be
        accomplished using the SUPER-USER community
        string which Norand has defined;
     2) This SUPER-USER community string, or password,
        is defined in the first row of the community
        table.  The communityName contained in the first
        row of the community table is always the
        SUPER-USER community string.  This community
        string (communityName) may be modified.
     3) All rows of the community table are modifiable
        (SET) when using the SUPER-USER community
        string.  However, for the first row of
        the community table, only the communityName
        object is modifiable. This ensures that the
        SUPER-USER will always have maximum access to
        the MIB data.  All other rows in the community
        Table are accessible as defined in the MIB
        definition.
     4) The SUPER-USER and other default community
        string values can be found in Norand's User's
        Guide."
     ::= { v1Config 2 }

-- Row Definition

 communityEntry        OBJECT-TYPE
     SYNTAX  CommunityEntry
     ACCESS  not-accessible
     STATUS  mandatory
     DESCRIPTION
     "Each entry relates to a specific community & associates to
     it access privileges."
     INDEX   { communityIndex }
     ::= {communityTable 1 }

-- Columnar Object Definition

 CommunityEntry ::=
     SEQUENCE {
         communityIndex          INTEGER,
         communityStatus         INTEGER,
         communityName           DisplayString,
         communityPrivileges     INTEGER
 }

 -- Leaf Definition

 communityIndex        OBJECT-TYPE
     SYNTAX   INTEGER
     ACCESS   read-only
     STATUS   mandatory
     DESCRIPTION
     "Identifies the community row"
     ::= { communityEntry 1 }
```

```
    communityStatus       OBJECT-TYPE
      SYNTAX  INTEGER {  enabled(1),
                        disabled(2),
                         deleted(3) }
      ACCESS  read-write
      STATUS  mandatory
      DESCRIPTION
      "Status of a community record.
      Alterations to the table may only be performed by
      a manager using the SUPER-USER community name.
      Status types:
             Enabled = Community record active
             Disabled = Community record not active
             Deleted = Disables and nulls objects in
             record"
      ::= { communityEntry 2 }

    communityName         OBJECT-TYPE
      SYNTAX  DisplayString (SIZE (0..15))
      ACCESS  read-write
      STATUS  mandatory
      DESCRIPTION
      "The authoritative name for the community.  Unless the
      Norand SUPER-USER community name is employed, a GET from
      this column yields an access violation."
      ::= { communityEntry 3 }

    communityPrivileges   OBJECT-TYPE
      SYNTAX  INTEGER {    get-only(1),
                        set-and-get(3) }
      ACCESS  read-write
      STATUS  mandatory
      DESCRIPTION
      "SET and GET privileges of community."
      ::= { communityEntry 4 }

-- Norand trap table defines all trap target IP addresses

   -- Table Definition

    trapTargetTable     OBJECT-TYPE
      SYNTAX  SEQUENCE OF TrapTargetEntry
      ACCESS  not-accessible
      STATUS  mandatory
      DESCRIPTION
      "The trap target table specifies the IP address of SNMPv1
      managers that expect trap notifications."
      ::= { v1Config 3 }

   -- Row Definition
```

```
 trapTargetEntry    OBJECT-TYPE
   SYNTAX  TrapTargetEntry
   ACCESS  not-accessible
   STATUS  mandatory
   DESCRIPTION
   "Each entry relates to a specific named manager at a given
   IP address & belonging to given community."
   INDEX   { trapTargetIndex }
   ::= {trapTargetTable 1 }
-- Columnar Object Definition
 TrapTargetEntry ::=
   SEQUENCE {
     trapTargetIndex     INTEGER,
     trapTargetStatus    INTEGER,
     trapTargetName      DisplayString,
     trapTargetIpAddress IpAddress
 }
-- Leaf Definition
 trapTargetIndex    OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
   "Identifies the trapTarget row"
   ::= { trapTargetEntry 1 }
 trapTargetStatus   OBJECT-TYPE
   SYNTAX  INTEGER {  enabled(1),
                     disabled(2),
                      deleted(3) }
   ACCESS  read-write
   STATUS  mandatory
   DESCRIPTION
   "Status of a trapTarget record."
   ::= { trapTargetEntry 2 }
 trapTargetName     OBJECT-TYPE
   SYNTAX  DisplayString (SIZE (0..16))
   ACCESS  read-write
   STATUS  mandatory
   DESCRIPTION
   "The authoritative name for the trapTarget."
   ::= { trapTargetEntry 3 }
 trapTargetIpAddress OBJECT-TYPE
   SYNTAX  IpAddress
   ACCESS  read-write
   STATUS  mandatory
   DESCRIPTION
   "IP Address of manager (which is assumed to be bound to &
   listening on port 162)."
   ::= { trapTargetEntry 4 }
```

```
nBridge                OBJECT IDENTIFIER ::= { norandNET 17 }

    rt                 OBJECT IDENTIFIER ::= { nBridge 2 }

        -- The RT Table

        -- Table Definition

        rtTable            OBJECT-TYPE
          SYNTAX SEQUENCE OF RTEntry
          ACCESS not-accessible
          STATUS mandatory
          DESCRIPTION
          "Each entry in this table provides routing information for
          child nodes which are reachable via a route."
          ::= { rt 2 }

        -- Row Definition

        rtEntry            OBJECT-TYPE
          SYNTAX RTEntry
          ACCESS not-accessible
          STATUS mandatory
          INDEX { rtDestination }
          ::= { rtTable 1 }

        -- Columnar Object Definition

        RTEntry ::=
        SEQUENCE {
            rtDestination    PhysAddress,
            rtPort           INTEGER,
            rtAge            INTEGER,
            rtNodeId         INTEGER,
            rtAttachId       INTEGER,
            rtAttachTime     TimeTicks,
            rtApEaddr        PhysAddress,
            rtHopAddrLen     INTEGER,
            rtHopAddr16      INTEGER,
            rtHopEaddr       PhysAddress,
            rtIsBound        INTEGER,
            rtIsRemote       INTEGER,
            rtIsChild        INTEGER,
            rtIsAp           INTEGER,
            rtIsDistributed  INTEGER,
            rtIsRemoteLan    INTEGER,
            rtNS             INTEGER,
            rtNR             INTEGER
        }

        -- Leaf Definition
```

```
rtDestination      OBJECT-TYPE
SYNTAX PhysAddress
ACCESS read-only
STATUS mandatory
DESCRIPTION
”The 802 address of the destination.”
::= { rtEntry 1 }

rtPort             OBJECT-TYPE
SYNTAX INTEGER (1..4)
ACCESS read-only
STATUS mandatory
DESCRIPTION
”The MAC-R port ID (1-4).  A number which uniquely identifies
the port.”
::= { rtEntry 2 }

rtAge              OBJECT-TYPE
SYNTAX INTEGER
ACCESS read-only
STATUS mandatory
DESCRIPTION
”The time (in minutes) since the route was updated.”
::= { rtEntry 3 }

rtNodeId           OBJECT-TYPE
SYNTAX INTEGER (0..65535)
ACCESS read-only
STATUS mandatory
DESCRIPTION
”16-bit node ID of the destination.  A 16-bit identifier which
uniquely identifies an OWL node in an OWL LAN.”
::= { rtEntry 4 }

rtAttachId         OBJECT-TYPE
SYNTAX INTEGER (0..65535)
ACCESS read-only
STATUS mandatory
DESCRIPTION
”Attach sequence number.  The sequence number is copied from
an OWL ATTACH request PDU.  The sequence number is not valid
for ’remote’ nodes.”
::= { rtEntry 5 }

rtAttachTime       OBJECT-TYPE
SYNTAX TimeTicks
ACCESS read-only
STATUS mandatory
DESCRIPTION
”Last attach time (.01 secs.)”
::= { rtEntry 6 }
```

```
rtApEaddr          OBJECT-TYPE
SYNTAX PhysAddress
ACCESS read-only
STATUS mandatory
DESCRIPTION
“802 address of AP which is the first hop on the path to the
destination”
::= { rtEntry 7 }

rtHopAddrLen       OBJECT-TYPE
SYNTAX INTEGER { twoByte(2),
                 sixByte(6) }
ACCESS read-only
STATUS mandatory
DESCRIPTION
”MAC-D address length (2 or 6). A MAC-D entity may use either
16-bit locally assigned addresses or 48-bit 802 addresses.”
::= { rtEntry 8 }

rtHopAddr16        OBJECT-TYPE
SYNTAX INTEGER (0..65535)
ACCESS read-only
STATUS mandatory
DESCRIPTION
”16-bit MAC-D address ( if rtHopAddrLen is twoByte(2) )”
::= { rtEntry 9 }

rtHopEaddr         OBJECT-TYPE
SYNTAX PhysAddress
ACCESS read-only
STATUS mandatory
DESCRIPTION
”48-bit MAC-D address ( if rtHopAddrLen is sixByte(6) )”
::= { rtEntry 10 }

rtIsBound          OBJECT-TYPE
SYNTAX INTEGER { true(1), false(2) }
ACCESS read-only
STATUS mandatory
DESCRIPTION
”True if the destination is fully attached and the path can be
used to forward data.”
::= { rtEntry 11 }

rtIsRemote         OBJECT-TYPE
SYNTAX INTEGER { true(1), false(2) }
ACCESS read-only
STATUS mandatory
DESCRIPTION
”True if the destination is a non-OWL node”
::= { rtEntry 12 }
```

```
          rtIsChild          OBJECT-TYPE
          SYNTAX INTEGER { true(1), false(2) }
          ACCESS read-only
          STATUS mandatory
          DESCRIPTION
          "True if the destination is a child node"
          ::= { rtEntry 13 }
          rtIsAp             OBJECT-TYPE
          SYNTAX INTEGER { true(1), false(2) }
          ACCESS read-only
          STATUS mandatory
          DESCRIPTION
          "True if the destination is an AP"
          ::= { rtEntry 14 }
          rtIsDistributed    OBJECT-TYPE
          SYNTAX INTEGER { true(1), false(2) }
          ACCESS read-only
          STATUS mandatory
          DESCRIPTION
          "True if the path is through a distributed AP (root node
          only)"
          ::= { rtEntry 15 }
          rtIsRemoteLan      OBJECT-TYPE
          SYNTAX INTEGER { true(1), false(2) }
          ACCESS read-only
          STATUS mandatory
          DESCRIPTION
          "True if the destination is a remote LAN"
          ::= { rtEntry 16 }
          rtNS               OBJECT-TYPE
          SYNTAX INTEGER (0..65535)
          ACCESS read-only
          STATUS mandatory
          DESCRIPTION
          "MAC-R send sequence number for terminal nodes.  The 16-bit
          (0-65535) sequence number of the last OWL data request PDU
          sent to the destination"
          ::= { rtEntry 17 }
          rtNR               OBJECT-TYPE
          SYNTAX INTEGER (0..65535)
          ACCESS read-only
          STATUS mandatory
          DESCRIPTION
          "MAC-R receive sequence number for terminal nodes.  The 16-bit
          (0-65535) sequence number of the last OWL data request PDU
          received from the destination"
          ::= { rtEntry 18 }
     brg                OBJECT IDENTIFIER ::= { nBridge 3 }
          -- The BRG Table
```

```
-- Table Definition
brgTable           OBJECT-TYPE
 SYNTAX SEQUENCE OF BRGEntry
 ACCESS not-accessible
 STATUS mandatory
 DESCRIPTION
 "Each entry in this table provides bridge information for
 child nodes which are reachable via a bridge."
 ::= { brg 2 }
-- Row Definition
brgEntry           OBJECT-TYPE
 SYNTAX BRGEntry
 ACCESS not-accessible
 STATUS mandatory
 INDEX { brgDestination }
 ::= { brgTable 1 }
-- Columnar Object Definition
BRGEntry ::=
 SEQUENCE {
   brgDestination PhysAddress,
   brgPort        INTEGER,
   brgAge         INTEGER,
   brgType        INTEGER,
   brgIsPermanent INTEGER,
   brgTimestamp   TimeTicks
}
-- Leaf Definition
brgDestination     OBJECT-TYPE
 SYNTAX PhysAddress
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 "The 802 address of the destination."
 ::= { brgEntry 1 }
brgPort            OBJECT-TYPE
 SYNTAX INTEGER (1..4)
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 "MAC-R port ID (1-4).  A number which uniquely identifies the
 port."
 ::= { brgEntry 2 }
brgAge             OBJECT-TYPE
 SYNTAX INTEGER
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 "Time (in minutes) since the entry was updated."
 ::= { brgEntry 3 }
```

```
    brgType            OBJECT-TYPE
     SYNTAX INTEGER {    primary(1),
                       secondary(2),
                        outbound(3),
                         inbound(4) }
     ACCESS read-only
     STATUS mandatory
     DESCRIPTION
     "Entry Type:
       primary   = 1,
       secondary = 2,
       outbound  = 3,
       inbound   = 4 "
     ::= { brgEntry 4 }

    brgIsPermanent     OBJECT-TYPE
     SYNTAX INTEGER { true(1), false(2) }
     ACCESS read-only
     STATUS mandatory
     DESCRIPTION
     "TRUE, if the entry is permanent."
     ::= { brgEntry 5 }

    brgTimestamp       OBJECT-TYPE
     SYNTAX TimeTicks
     ACCESS read-only
     STATUS mandatory
     DESCRIPTION
     "The time when the primary or inbound entry was added or the
     time when the secondary entry was added or re-attached."
     ::= { brgEntry 6 }

addr               OBJECT IDENTIFIER ::= { nBridge 4 }

    -- The Addr Table

    -- Table Definition

    addrTable          OBJECT-TYPE
     SYNTAX SEQUENCE OF AddrEntry
     ACCESS not-accessible
     STATUS mandatory
     DESCRIPTION
     "Each entry in this table provides address information for
     all OWL nodes in the network.  This table exists only at the
     root node."
     ::= { addr 2 }

    -- Row Definition

    addrEntry          OBJECT-TYPE
     SYNTAX AddrEntry
     ACCESS not-accessible
     STATUS mandatory
     INDEX { addrDestination }
     ::= { addrTable 1 }
```

```
-- Columnar Object Definition

AddrEntry ::=
 SEQUENCE {
   addrDestination PhysAddress,
   addrAge         INTEGER,
   addrNodeId      INTEGER,
   addrAlias       DisplayString,
   addrDeviceId    INTEGER,
   addrIpAddress   IpAddress
}

-- Leaf Definition

addrDestination    OBJECT-TYPE
 SYNTAX PhysAddress
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”The 802 address of the registered port.”
 ::= { addrEntry 1 }

addrAge            OBJECT-TYPE
 SYNTAX INTEGER
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”The time (in minutes) since the entry was updated.”
 ::= { addrEntry 2 }

addrNodeId         OBJECT-TYPE
 SYNTAX INTEGER (0..65535)
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”16-bit (0-65535) node/port ID”
 ::= { addrEntry 3 }

addrAlias          OBJECT-TYPE
 SYNTAX DisplayString (SIZE (0..16))
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”An alias for the 802 address”
 ::= { addrEntry 4 }

addrDeviceId       OBJECT-TYPE
 SYNTAX INTEGER (0..65535)
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”Device ID (0-65535).  An OWL node can, optionally, set a
 device ID in a registration request PDU.”
 ::= { addrEntry 5 }
```

```
    addrIpAddress      OBJECT-TYPE
     SYNTAX IpAddress
     ACCESS read-only
     STATUS mandatory
     DESCRIPTION
     "32-bit IP address for IP nodes (e.g. APs)"
     ::= { addrEntry 6 }

brgState           OBJECT IDENTIFIER ::= { nBridge 6 }

    -- The Bridge State Group

    bsAddress                OBJECT-TYPE
      SYNTAX PhysAddress
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      "802 address of the AP"
      ::= { brgState 3 }

    bsLanId                  OBJECT-TYPE
      SYNTAX INTEGER (0..254)
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      "OWL LAN ID (0-254)"
      ::= { brgState 4 }

    bsCostToRoot             OBJECT-TYPE
      SYNTAX INTEGER
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      "Path cost to the root"
      ::= { brgState 5 }

    bsIsRoot                 OBJECT-TYPE
      SYNTAX INTEGER { true(1), false(2) }
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      "TRUE, if the AP is the root"
      ::= { brgState 6 }

    bsIsAttached             OBJECT-TYPE
      SYNTAX INTEGER { true(1), false(2) }
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      "TRUE, if the AP is attached"
      ::= { brgState 7 }
```

```
bsAttachId              OBJECT-TYPE
  SYNTAX INTEGER (0..65535)
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”16-bit (0-65535) ATTACH sequence number.  This number is
  incremented each time the AP sends an ATTACH request.”
  ::= { brgState 8 }

bsMyRootPriority        OBJECT-TYPE
  SYNTAX INTEGER (0..7)
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Root priority of the AP (0-7).  An AP with a root priority
  of 0 can not become the root node.  The AP with the highest
  priority will become the root in an OWL LAN”
  ::= { brgState 9 }

bsRootPort              OBJECT-TYPE
  SYNTAX INTEGER (1..4)
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”MAC-R root port number. The port number (1-4) of the port
  used to communicate with the parent node.”
  ::= { brgState 10 }

bsDesignatedRootAddress  OBJECT-TYPE
  SYNTAX PhysAddress
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”802 address of the current root”
  ::= { brgState 11 }

bsDesignatedRootPriority OBJECT-TYPE
  SYNTAX INTEGER (1..7)
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Root priority of the current root (1-7)”
  ::= { brgState 12 }

bsDesignatedRootSequence OBJECT-TYPE
  SYNTAX INTEGER (0..255)
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Root sequence of the current root (0-255).  The sequence
  number identifies a single instance of the root.”
  ::= { brgState 13 }
```

```
bsParentAddress          OBJECT-TYPE
  SYNTAX PhysAddress
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”802 address of the parent AP”
  ::= { brgState 14 }

bsPortCount              OBJECT-TYPE
  SYNTAX INTEGER
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Number of MAC-R ports”
  ::= { brgState 15 }

bsNodeId                 OBJECT-TYPE
  SYNTAX INTEGER (0..65535)
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”16-bit node ID (0-65535).  The node ID uniquely identifies
  the node in an OWL LAN.”
  ::= { brgState 16 }

bsRootChangedCount       OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Number of times that the root has changed.”
  ::= { brgState 17 }

bsRootCount              OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Number of times that the AP became the root”
  ::= { brgState 18 }

bsAttachCount            OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Number of times that the AP has changed from an unattached
  state to an attached state.”
  ::= { brgState 19 }
```

```
bsDetachReason          OBJECT-TYPE
  SYNTAX INTEGER
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Last detach reason code.  The code indicates the
  reason that the AP became unattached for the last
  occurrence.
  0 - Initial Value,
  1 - A new root node was detected,
  2 - The network inactivity timer expired,
  4 - A better path to the root was detected,
  5 - The node’s parent became unattached,
  7 - The node was in a detach list in a HELLO PDU,
  8 - The node was functioning as the root and
      relinquished the root status,
  9 - The maximum number of attache retries was
      exceeded without receiving an ATTACH response
      PDU, 900-90F - A MAC-D link error occurred
      while sending a PDU to the parent node.”
  ::= { brgState 20 }

bsNetworkTime           OBJECT-TYPE
  SYNTAX TimeTicks
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Distributed network time (.01 sec)”
  ::= { brgState 21 }

bsUniFloodLevel         OBJECT-TYPE
  SYNTAX INTEGER (1..2)
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Flooding level for unicast frames (1-2)”
  ::= { brgState 22 }

bsMultiFloodLevel       OBJECT-TYPE
  SYNTAX INTEGER (0..3)
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Flooding level for multicast frames (0-3)”
  ::= { brgState 23 }

bsIsPrimaryBridge       OBJECT-TYPE
  SYNTAX INTEGER { true(1), false(2) }
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”TRUE, if the AP bridges to the distribution LAN”
  ::= { brgState 24 }
```

```
    bsIsSecondaryBridge      OBJECT-TYPE
      SYNTAX INTEGER { true(1), false(2) }
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      ”TRUE, if the AP is the designated bridge for a secondary
      LAN”
      ::= { brgState 25 }

    bsUniFilterExpr          OBJECT-TYPE
      SYNTAX INTEGER (0..255)
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      ”Unicast ethernet filter ID (0-255).  If the ID is non-zero,
      it points to a user defined expression which is used to
      filter unicast frames on the ethernet port.”
      ::= { brgState 26 }

    bsMultiFilterExpr        OBJECT-TYPE
      SYNTAX INTEGER (0..255)
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      ”Multicast ethernet filter ID (0-255). If the ID is
      non-zero, it points to a user defined expression which is
      used to filter multicast frames on the ethernet port.”
      ::= { brgState 27 }

bridgeStats        OBJECT IDENTIFIER ::= { nBridge 7 }

    bstcRouteCount          OBJECT-TYPE
      SYNTAX Gauge
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      ”Total route table entries”
      ::= { bridgeStats 3 }

    bstcChildCount          OBJECT-TYPE
      SYNTAX Gauge
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      ”Number of attached children”
      ::= { bridgeStats 4 }

    bstcChildApCount        OBJECT-TYPE
      SYNTAX Gauge
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      ”Number of attached AP children”
      ::= { bridgeStats 5 }
```

```
bstcRemoteCount         OBJECT-TYPE
  SYNTAX Gauge
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Non-OWL bridge table entries”
  ::= { bridgeStats 6 }

bstcPrimaryCount        OBJECT-TYPE
  SYNTAX Gauge
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Distribution LAN bridge table entries”
  ::= { bridgeStats 7 }

bstcInboundCount        OBJECT-TYPE
  SYNTAX Gauge
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Inbound bridge table entries”
  ::= { bridgeStats 8 }

bstcSecondaryCount      OBJECT-TYPE
  SYNTAX Gauge
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Secondary LAN bridge table entries”
  ::= { bridgeStats 9 }

bstcRemoteLanCount      OBJECT-TYPE
  SYNTAX Gauge
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Route entries for remote LANs”
  ::= { bridgeStats 10 }

bstcRouteGetErrors      OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Route table overflow errors”
  ::= { bridgeStats 11 }

bstcEntryGetErrors      OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Bridge table overflow errors”
  ::= { bridgeStats 12 }
```

```
            bstcRmtLanGetErrors     OBJECT-TYPE
              SYNTAX Counter
              ACCESS read-only
              STATUS mandatory
              DESCRIPTION
              "Remote LAN overflow errors"
              ::= { bridgeStats 13 }

            bstcRouteSeqErrors      OBJECT-TYPE
              SYNTAX Counter
              ACCESS read-only
              STATUS mandatory
              DESCRIPTION
              "Out-of-sequence route update errors"
              ::= { bridgeStats 14 }

            bstcDeleteSeqErrors     OBJECT-TYPE
              SYNTAX Counter
              ACCESS read-only
              STATUS mandatory
              DESCRIPTION
              "Out-of-sequence route delete errors"
              ::= { bridgeStats 15 }

            bstcEntrySeqErrors      OBJECT-TYPE
              SYNTAX Counter
              ACCESS read-only
              STATUS mandatory
              DESCRIPTION
              "Out-of-sequence bridge entry update errors"
              ::= { bridgeStats 16 }

            bstcInvalidUpdateErrors OBJECT-TYPE
              SYNTAX Counter
              ACCESS read-only
              STATUS mandatory
              DESCRIPTION
              "Invalid route update errors"
              ::= { bridgeStats 17 }

    nTransport              OBJECT IDENTIFIER ::= { norandNET 102 }

         wst                OBJECT IDENTIFIER ::= { nTransport 2 }

            wstReset                OBJECT-TYPE
              SYNTAX INTEGER { true(1), false(2) }
              ACCESS read-write
              STATUS mandatory
              DESCRIPTION
              "Reset variable for the wst group"
              ::= { wst 1 }
```

```
wstLastReset            OBJECT-TYPE
  SYNTAX TimeTicks
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Elapsed time since the wst group was last reset”
  ::= { wst 2 }

wstClockTicks           OBJECT-TYPE
  SYNTAX TimeTicks
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”WTP clock ticks in hundredths of seconds.”
  ::= { wst 3 }

wstInternalErrorCount   OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Number of internal WTP errors”
  ::= { wst 4 }

wstSessionsLost         OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Number of active WTP connections which were
  terminated.”
  ::= { wst 5 }

wstFrmrCount            OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Number of WTP Frame Rejects which were
  transmitted”
  ::= { wst 6 }

wstDataBytesXmit        OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Total number of bytes transmitted.”
  ::= { wst 7 }
```

```
wstDataBytesRecv        OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Total number of bytes received”
  ::= { wst 8 }

wstIframesXmit          OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Total number of WTP data frames transmitted.”
  ::= { wst 9 }

wstIframesRecv          OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Total number of WTP data frames received.”
  ::= { wst 10 }

wstIframesRetransmitted OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Total number of WTP data frames which were lost and
  retransmitted.”
  ::= { wst 11 }

wstCheckSumErrors       OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The number of WTP information frames which were discarded
  to WTP check sum errors.”
  ::= { wst 12 }

wstHlDiscardCount       OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The number of received WTP information frames which were
  discarded without being read by a higher layer application.
  Received information frames are queued per connection and
  should normally not be discarded.”
  ::= { wst 13 }
```

```
wstState                OBJECT-TYPE
  SYNTAX INTEGER (0..2147483647)
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The internal software state of the WTP module.”
  ::= { wst 14 }

wstBrgBusy              OBJECT-TYPE
  SYNTAX INTEGER (0..1)
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The value of wstBrgBusy is 1 if the WTP currently has an
  outstanding network transmit request.  The value is
  otherwise 0.”
  ::= { wst 15 }

wstT1Timeouts           OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The total number of times the WTP T1 timer has expired
  (i.e. due to lost frames).”
  ::= { wst 16 }

wstT1Bucket1            OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The total number of times that 2 consecutive T1 errors have
  occurred.”
  ::= { wst 17 }

wstT1Bucket2            OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The total number of times that 3 consecutive T1 errors have
  occurred.”
  ::= { wst 18 }

wstT1Bucket3            OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The total number of times that 5 consecutive T1 errors have
  occurred.”
  ::= { wst 19 }
```

```
wstT1Bucket4            OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The total number of times that 6 or more consecutive T1
  errors have occurred.”
  ::= { wst 20 }

wstSABMCount            OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The total SABM frames received by the local WTP entity.”
  ::= { wst 21 }

wstDiscCount            OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The number of DISC frames received by the local WTP
  entity.”
  ::= { wst 22 }

wstDMCount              OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The number of DM frames received by the local WTP entity.”
  ::= { wst 23 }

wstRNRCount             OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The number of RNR frames received by the local WTP entity.”
  ::= { wst 24 }

wstRejectCount          OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The number of REJECT frames received by the local WTP
  entity.”
  ::= { wst 25 }
```

```
wstSessionsReset        OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  "The number of active WTP sessions which have been reset in
  the local WTP entity."
  ::= { wst 26 }

wstBrgSrvTime           OBJECT-TYPE
  SYNTAX TimeTicks
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  "The total bridge service time for WTP transmissions in
  hundredths of seconds."
  ::= { wst 27 }

wstBrgSrvCount          OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  "The total number of WTP bridge transmissions.  The average
  service time is wstBrgSrvTime/wstBrgSrvCount."
  ::= { wst 28 }

wstBrgSrvThreshold      OBJECT-TYPE
  SYNTAX TimeTicks
  ACCESS read-write
  STATUS mandatory
  DESCRIPTION
  "A threshold bridge service time for WTP transmissions.
  Service times which exceed the threshold are counted in
  wstBrgSrvThreshCount."
  ::= { wst 29 }

wstBrgSrvThreshCount    OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  "A count of the number of times the bridge service time has
  exceeded the threshold time defined by wstBrgSrvThreshold."
  ::= { wst 30 }

wstBrgSrvLongest        OBJECT-TYPE
  SYNTAX TimeTicks
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  "The longest network service time measured for an WTP
  transmission."
  ::= { wst 31 }
```

```
    wstBrgTxErrors          OBJECT-TYPE
      SYNTAX Counter
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      ”The number of transmission errors reported to the WTP layer
      by the network layer.”
      ::= { wst 32 }

    wstFatalBrgErrors       OBJECT-TYPE
      SYNTAX Counter
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      ”The number of fata errors reported to the WTP layer by the
      network layer.  A fatal error indicates that the network
      root node has changed.”
      ::= { wst 33 }

hlit               OBJECT IDENTIFIER ::= { nTransport 3 }

    hlitReset       OBJECT-TYPE
      SYNTAX INTEGER { true(1), false(2) }
      ACCESS read-write
      STATUS mandatory
      DESCRIPTION
      ”Reset variable for the HLIT group”
      ::= { hlit 1 }

    hlitLastReset   OBJECT-TYPE
      SYNTAX TimeTicks
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      ”Elapsed time since the HLIT group was reset”
      ::= { hlit 2 }

    hlitTime        OBJECT-TYPE
      SYNTAX TimeTicks
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      ”This is the cumulative time that the WTP took to service
      transmit requests.  Time is in hundredths of seconds.”
      ::= { hlit 3 }

    hlitCount       OBJECT-TYPE
      SYNTAX Counter
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      ”This is the number of times the WTP was sent a transmit
      request.”
      ::= { hlit 4 }
    hlitThreshold   OBJECT-TYPE
      SYNTAX TimeTicks
      ACCESS read-write
      STATUS mandatory
      DESCRIPTION
      "When the WTP exceeds this time while servicing a transmit
      request, the hlitThreshCount will be incremented.  Time is
      in hundredths of seconds."
      ::= { hlit 5 }

    hlitThreshCount OBJECT-TYPE
      SYNTAX Counter
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      "When the WTP exceeds the time in hlitThreshold, this count
      is incremented.  Time in hundreths of seconds."
      ::= { hlit 6 }

    hlitLongest     OBJECT-TYPE
      SYNTAX Counter
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      "This is the maximum time that the WTP took to service
      transmit requests.  Time is in hundreths of seconds."
      ::= { hlit 7 }

wcbst              OBJECT IDENTIFIER ::= { nTransport 4 }

    wcbstReset                 OBJECT-TYPE
      SYNTAX INTEGER { true(1), false(2) }
      ACCESS read-write
      STATUS mandatory
      DESCRIPTION
      "Reset variable for the wcbs table"
      ::= { wcbst 1 }

    wcbstLastReset             OBJECT-TYPE
      SYNTAX TimeTicks
      ACCESS read-only
      STATUS mandatory
      DESCRIPTION
      "Elapsed time since the wcbs table was reset"
      ::= { wcbst 2 }

    wcbsTable                  OBJECT-TYPE
      SYNTAX SEQUENCE OF WcbsEntry
      ACCESS not-accessible
      STATUS mandatory
      DESCRIPTION
      "WTP connection control block statistics table.  Each entry
      is for an WTP connection statistics object.  Statistics are
      per WTP connection."
      ::= { wcbst 3 }
```

```
wcbsEntry                OBJECT-TYPE
  SYNTAX WcbsEntry
  ACCESS not-accessible
  STATUS mandatory
  INDEX { wcbstIndex }
  ::= { wcbsTable 1 }

WcbsEntry ::=
  SEQUENCE {
    wcbstIndex                INTEGER,
    wcbstInUse                INTEGER,
    wcbstHlState              INTEGER,
    wcbstDsap                 INTEGER,
    wcbstSsap                 INTEGER,
    wcbstNet16Addr            INTEGER,
    wcbstT1Value              INTEGER,
    wcbstT1Average            INTEGER,
    wcbstT1Deviation          INTEGER,
    wcbstT1Timeouts           Counter,
    wcbstT2Timeouts           Counter,
    wcbstTiTimeouts           Counter,
    wcbstFramesRecv           Counter,
    wcbstIframesRecv          Counter,
    wcbstIframesDiscarded     Counter,
    wcbstDataBytesRecv        Counter,
    wcbstFramesXmit           Counter,
    wcbstIframesXmit          Counter,
    wcbstDataBytesXmit        Counter,
    wcbstIframesRetransmitted Counter,
    wcbstState                INTEGER
  }

wcbstIndex               OBJECT-TYPE
  SYNTAX INTEGER (1..999)
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Connection control block identifier”
  ::= { wcbsEntry 1 }

wcbstInUse               OBJECT-TYPE
  SYNTAX INTEGER (0..1)
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”This variable is 1 if the control block is in use.”
  ::= { wcbsEntry 2 }
```

```
wcbstHlState            OBJECT-TYPE
  SYNTAX INTEGER (0..9999)
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The control block state.  0=closed, 1=busy, 2=connect,
  3=listen, 4=unitdata, 5=unitdata listen, 6=open, 7=send,
  8=closing, 9=disconnect.”
  ::= { wcbsEntry 3 }

wcbstDsap               OBJECT-TYPE
  SYNTAX INTEGER (0..255)
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Destination service access point.”
  ::= { wcbsEntry 4 }

wcbstSsap               OBJECT-TYPE
  SYNTAX INTEGER (0..255)
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Source service access point”
  ::= { wcbsEntry 5 }

wcbstNet16Addr          OBJECT-TYPE
  SYNTAX INTEGER (2048..9192)
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The 16-bit network address of the remote node which is
  active on the connection, if the control block is in an
  active state.”
  ::= { wcbsEntry 6 }

wcbstT1Value            OBJECT-TYPE
  SYNTAX INTEGER (0..999)
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The current response time-out value in tenths of seconds.”
  ::= { wcbsEntry 7 }

wcbstT1Average          OBJECT-TYPE
  SYNTAX INTEGER (0..999)
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The average response time-out value in tenths of seconds.”
  ::= { wcbsEntry 8 }
```

```
wcbstT1Deviation          OBJECT-TYPE
  SYNTAX INTEGER (0..999)
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”The average deviation from the T1Average values in tenths
  of seconds.”
  ::= { wcbsEntry 9 }

wcbstT1Timeouts           OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Number of T1 timeouts.  A T1 timeout occures when an
  expected response is lost.”
  ::= { wcbsEntry 10 }

wcbstT2Timeouts           OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Number of T2 timeouts.  A T2 timeout occures if the higher
  layer does not have data ready to send before an WTP
  supervisory response is required.”
  ::= { wcbsEntry 11 }

wcbstTiTimeouts           OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Number of Ti timeouts.  A Ti timeout occures when a
  connection is idle for an INACITVE TIMEOUT period.”
  ::= { wcbsEntry 12 }

wcbstFramesRecv           OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Total number of WTP frames received.”
  ::= { wcbsEntry 13 }

wcbstIframesRecv          OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Total number of WTP data frames received.”
  ::= { wcbsEntry 14 }
```

```
wcbstIframesDiscarded     OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Total number of WTP data frames which were discarded as
  duplicates.”
  ::= { wcbsEntry 15 }

wcbstDataBytesRecv        OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Total number of data bytes received.”
  ::= { wcbsEntry 16 }

wcbstFramesXmit           OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Total number of WTP frames transmitted.”
  ::= { wcbsEntry 17 }

wcbstIframesXmit          OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Total number of WTP data frames transmitted.”
  ::= { wcbsEntry 18 }

wcbstDataBytesXmit        OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Total number of WTP data frames transmitted.”
  ::= { wcbsEntry 19 }

wcbstIframesRetransmitted OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Total number of data frames which were lost and had to be
  retransmitted.”
  ::= { wcbsEntry 20 }
```

```
         wcbstState              OBJECT-TYPE
           SYNTAX INTEGER (0..2147483647)
           ACCESS read-only
           STATUS mandatory
           DESCRIPTION
           ”The internal software state of the LLC control block.”
           ::= { wcbsEntry 21 }
nApplication          OBJECT IDENTIFIER ::= { norandNET 104 }
       gwSession      OBJECT IDENTIFIER ::= { nApplication 1 }
             gs         OBJECT IDENTIFIER ::= { gwSession 1 }
                 gsReset              OBJECT-TYPE
                  SYNTAX INTEGER { true(1), false(2) }
                  ACCESS read-write
                  STATUS mandatory
                  DESCRIPTION
                  ”Reset variable for the gs group”
                  ::= { gs 1 }
                 gsLastReset          OBJECT-TYPE
                  SYNTAX TimeTicks
                  ACCESS read-only
                  STATUS mandatory
                  DESCRIPTION
                  ”The time elapsed since the last reset of gs group”
                  ::= { gs 2 }
                 -- The Gateway Statistics group
                 gsClockTicks         OBJECT-TYPE
                  SYNTAX TimeTicks
                  ACCESS read-only --read-reset
                  STATUS mandatory
                  DESCRIPTION
                  ”Number of .01 second timer ticks since the timer was
                  reset.”
                  ::= { gs 3 }
                 gsNetworkErrors      OBJECT-TYPE
                  SYNTAX Counter
                  ACCESS read-only --read-reset
                  STATUS mandatory
                  DESCRIPTION
                  ”Number of times the network has been lost.”
                  ::= { gs 4 }
                 gsHostDataCount      OBJECT-TYPE
                  SYNTAX Counter
                  ACCESS read-only --read-reset
                  STATUS mandatory
                  DESCRIPTION
                  ”Number of data blocks received from the host
                  computer.”
                  ::= { gs 5 }
```

```
gsHostHaltCount      OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”Number of HALT commands received from the host
 task.”
 ::= { gs 6 }

gsHostInvalidCount   OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”Number of invalid messages received from the host
 task.”
 ::= { gs 7 }

gsHostDiscardCount   OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”Number of blocks from the host computer which were
 discarded.”
 ::= { gs 8 }

gsConnectCount       OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”The total number of transport connections.”
 ::= { gs 9 }

gsCloseCount         OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”Number of times a transport connection was closed.”
 ::= { gs 10 }

gsBlocksXmit         OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”Number of blocks passed to the network”
 ::= { gs 11 }
```

```
gsDataXmit          OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”Number of data blocks passed to the network”
 ::= { gs 12 }
gsBlocksRecv        OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”Number of blocks received from the network”
 ::= { gs 13 }
gsDataRecv          OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”Number of data blocks received from the network.”
 ::= { gs 14 }
gsTermResetCount    OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”Number of times a session was reset by a terminal”
 ::= { gs 15 }
gsTermContinueCount OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”Number of times a session was continued
 after a transport connection was lost.”
 ::= { gs 16 }
gsTermInvalidCount  OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”Number of invalid blocks received from terminals.”
 ::= { gs 17 }
gsConnectionErrors  OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”Number of transport connection errors.”
 ::= { gs 18 }
```

```
gsConnectionResets   OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 "Number of times an active transport connection was
 reset."
 ::= { gs 19 }

gsRetransmissions    OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 "Number of data blocks which were retransmitted."
 ::= { gs 20 }

gsRecvSeqErrors      OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 "Number of data blocks which were discarded due to
 receive sequence errors."
 ::= { gs 21 }

gsXmitErrors         OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 "Number of transport transmission errors."
 ::= { gs 22 }

gsTermDiscardCount   OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 "Number of discarded blocks received from terminals."
 ::= { gs 23 }

gsHostDelayMax       OBJECT-TYPE
 SYNTAX INTEGER (0..9999)
 ACCESS read-write
 STATUS mandatory
 DESCRIPTION
 "The maximum host delay time. Host delay is the time
 a message is received from a terminal until the host
 responds. Times less than the maximum host delay time
 are counted as transactions times. Host delay time is
 measured in hundredths of seconds."
 ::= { gs 24 }
```

```
gsHostDelayTotal     OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”The total host delay time in hundredths of seconds,
 not including delay times which exceeded the maximum
 host delay.”
 ::= { gs 25 }

gsHostTransactions   OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”The total number of times the host responded to a
 terminal message in a time less than the maximum host
 delay time.”
 ::= { gs 26 }

gsRecvErrors         OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”The number of transport layer receive errors.”
 ::= { gs 27 }

gsTermDelayMax       OBJECT-TYPE
 SYNTAX INTEGER
 ACCESS read-write
 STATUS mandatory
 DESCRIPTION
 ”The maximum terminal transaction time in hundredths
 of seconds. Round-trip times which exceed the maximum
 delay are not included as terminal transactions.”
 ::= { gs 28 }

gsTermDelayThreshold OBJECT-TYPE
 SYNTAX INTEGER
 ACCESS read-write
 STATUS mandatory
 DESCRIPTION
 ”Terminal transactions which take longer than the
 value set in gsTermDelayThreshold are counted in
 gsTermThresholdCount. This threshold is specified in
 1/100th seconds.”
 ::= { gs 29 }
```

```
gsTermDelayTotal     OBJECT-TYPE
 SYNTAX TimeTicks
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”The total time for all terminal transactions.”
 ::= { gs 30 }

gsTermTransactions   OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”The total number of terminal transactions.  Terminal
 transaction information is sent to the controller
 from terminals.”
 ::= { gs 31 }

gsTermThresholdCount OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”The number of terminal transactions which exceed the
 threshold value set in gsTermDelayThreshold.”
 ::= { gs 32 }

gsTermMaxCount       OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”The number of terminal transactions which exceed the
 maximum transaction time set in gsTermDelayMax.”
 ::= { gs 33 }

gsHostMaxCount       OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”The number of transactions which exceed the maximum
 host delay time set in gsHostDelayMax.”
 ::= { gs 34 }

gsTermDelayTraceOn   OBJECT-TYPE
 SYNTAX INTEGER { true(1), false(2) }
 ACCESS read-write
 STATUS mandatory
 DESCRIPTION
 ”Terminal transaction delay diagnostics are enabled
 by setting gsTermDelayTraceOn to a non-zero value.
 The results of this trace are found in the tdTable.”
 ::= { gs 35 }
```

```
    gsHostActive         OBJECT-TYPE
     SYNTAX INTEGER { true(1), false(2) }
     ACCESS read-only
     STATUS mandatory
     DESCRIPTION
     ”Indicates whether the host is active and
     communicating with the controller. A value of 1 =
     active, a value of 0 = inactive. The default is 0.
     Once the host has communicated with the controller,
     this value will always show ’active’ (1) UNLESS the
     cpctPortInactTimeout value in the cpcTable is set to
     a non-zero value. If the controller has not received
     a response from the host in the time (in seconds)
     specified by cpctPortInactTimeout, gsHostActive will
     get set to 0.”
     ::= { gs 36 }

rcbs       OBJECT IDENTIFIER ::= { gwSession 2 }

    -- The RCBS Table

    rcbsTableReset         OBJECT-TYPE
     SYNTAX INTEGER { true(1), false(2) }
     ACCESS read-write
     STATUS mandatory
     DESCRIPTION
     ”Reset variable for the rcbsTable table”
     ::= { rcbs 2 }

    rcbsTableLastReset     OBJECT-TYPE
     SYNTAX TimeTicks
     ACCESS read-only
     STATUS mandatory
     DESCRIPTION
     ”The time elapsed since
     the last reset of rcbsTable table”
     ::= { rcbs 3 }

    -- Table Definition

    rcbsTable              OBJECT-TYPE
     SYNTAX SEQUENCE OF RcbsEntry
     ACCESS not-accessible
     STATUS mandatory
     DESCRIPTION
     ”SESSION control block status table.  Each entry
     corresponds to a terminal session.”
     ::= { rcbs 4 }

    -- Row Definition

    rcbsEntry              OBJECT-TYPE
     SYNTAX RcbsEntry
     ACCESS not-accessible
     STATUS mandatory
     INDEX { rcbsIndex }
     ::= { rcbsTable 1 }
```

```
-- Columnar Object Definition

RcbsEntry ::=
 SEQUENCE {
    rcbsIndex              INTEGER,
    rcbsTerminal           INTEGER,
    rcbsType               INTEGER,
    rcbsState              INTEGER,
    rcbsLLCIndex           INTEGER,
    rcbsHostDataCount      Counter,
    rcbsBlocksXmit         Counter,
    rcbsDataXmit           Counter,
    rcbsBlocksRecv         Counter,
    rcbsDataRecv           Counter,
    rcbsTermResetCount     Counter,
    rcbsTermContinueCount  Counter,
    rcbsCloseCount         Counter,
    rcbsHostTransactions   Counter,
    rcbsHostDelayTotal     Counter,
    rcbsHostDelayLast      INTEGER,
    rcbsTermResetTime      INTEGER,
    rcbsTermInactTime      INTEGER,
    rcbsHostInactTime      INTEGER,
    rcbsConnectionErrors   Counter,
    rcbsNetAddress         PhysAddress,
    rcbsSeqErrors          Counter,
    rcbsTermTransactions   Counter,
    rcbsTermDelayTotal     TimeTicks,
    rcbsTermDelayLast      INTEGER,
    rcbsTermDelayLongest   INTEGER,
    rcbsTermThresholdCount Counter,
    rcbsTermMaxCount       Counter,
    rcbsTermDelayTraceOn   INTEGER,
    rcbsHostAlias          DisplayString
}

-- Leaf Definition

rcbsIndex              OBJECT-TYPE
 SYNTAX INTEGER (1..65535)
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”Non zero table index.”
 ::= { rcbsEntry 1 }

rcbsTerminal           OBJECT-TYPE
 SYNTAX INTEGER (0..127)
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”Terminal session identifier”
 ::= { rcbsEntry 2 }
```

```
rcbsType               OBJECT-TYPE
 SYNTAX INTEGER (0..255)
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”The terminal power up type as passed by the terminal
 at the beginning of an RTC session.  (Unused until
 V4.16 of terminal software).  Values: 11, 21, 46, 47,
 61 indicate 3270 emulation; 62 = 5250 emulation; 63 =
 VT220; Any other value indicates Native mode.”
 ::= { rcbsEntry 3 }

rcbsState              OBJECT-TYPE
 SYNTAX INTEGER (0..99)
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”Session state. 0=reset state, 1=active state.”
 ::= { rcbsEntry 4 }

rcbsLLCIndex           OBJECT-TYPE
 SYNTAX INTEGER (0..65535)
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”Transport connection control block identifier.
 Corresponds to an lcbstIndex value.”
 ::= { rcbsEntry 5 }

rcbsHostDataCount      OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”Number of data blocks received from the host.”
 ::= { rcbsEntry 6 }

rcbsBlocksXmit         OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”Total number of blocks passed to the network.”
 ::= { rcbsEntry 7 }

rcbsDataXmit           OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”Number of data blocks passed to the network.”
 ::= { rcbsEntry 8 }
```

```
rcbsBlocksRecv          OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”Total number of blocks received from the terminal”
 ::= { rcbsEntry 9 }

rcbsDataRecv            OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”Number of data blocks received from the terminal.”
 ::= { rcbsEntry 10 }

rcbsTermResetCount      OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”Number of times the terminal has reset the session.”
 ::= { rcbsEntry 11 }

rcbsTermContinueCount  OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”Number of times a terminal has continued a session
 after the transport connection was lost and
 re-established.”
 ::= { rcbsEntry 12 }

rcbsCloseCount          OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”Number of times an active transport connection used
 by the terminal session was closed.”
 ::= { rcbsEntry 13 }

rcbsHostTransactions    OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”The total number of times the host responded to a
 message from the terminal in a time less than the
 maximum host delay time.”
 ::= { rcbsEntry 14 }
```

```
rcbsHostDelayTotal     OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”The total host delay for the terminal in hundredths
 of seconds, not including times which exceeded the
 maximum host delay time.”
 ::= { rcbsEntry 15 }

rcbsHostDelayLast      OBJECT-TYPE
 SYNTAX INTEGER
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”The host response time, in hundredths of seconds,
 for the last message sent from the terminal.”
 ::= { rcbsEntry 16 }

rcbsTermResetTime      OBJECT-TYPE
 SYNTAX INTEGER (0..2147483647)
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”The time, in seconds, since the terminal reset its
 session.”
 ::= { rcbsEntry 17 }

rcbsTermInactTime      OBJECT-TYPE
 SYNTAX INTEGER (0..2147483647)
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”The time, in seconds, since a message was received
 from the terminal.”
 ::= { rcbsEntry 18 }

rcbsHostInactTime      OBJECT-TYPE
 SYNTAX INTEGER (0..2147483647)
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”The time, in seconds, since a message was received
 from the host for the terminal.”
 ::= { rcbsEntry 19 }

rcbsConnectionErrors   OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”The number of times the terminal session lost a
 transport connection.”
 ::= { rcbsEntry 20 }
```

```
rcbsNetAddress        OBJECT-TYPE
 SYNTAX PhysAddress
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”The network address of the remote terminal defined
 by rcbsTerminal.”
 ::= { rcbsEntry 21 }

rcbsSeqErrors         OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”The number of messages discarded due to receive
 sequence errors.”
 ::= { rcbsEntry 22 }

rcbsTermTransactions  OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”The total number of terminal transactions
 for the terminal defined by rcbsTerminal.”
 ::= { rcbsEntry 23 }

rcbsTermDelayTotal    OBJECT-TYPE
 SYNTAX TimeTicks
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”The total time for all transactions for the
 terminal defined by rcbsTerminal.”
 ::= { rcbsEntry 24 }

rcbsTermDelayLast     OBJECT-TYPE
 SYNTAX INTEGER
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”The time for the last transaction for the terminal
 defined by rcbsTerminal, in 1/100th seconds.”
 ::= { rcbsEntry 25 }

rcbsTermDelayLongest  OBJECT-TYPE
 SYNTAX INTEGER
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”The time for the longest transaction for the
 terminal defined by rcbsTerminal, in 1/100th
 seconds.”
 ::= { rcbsEntry 26 }
```

```
    rcbsTermThresholdCount OBJECT-TYPE
     SYNTAX Counter
     ACCESS read-only --read-reset
     STATUS mandatory
     DESCRIPTION
     ”The total number of transactions which exceeded the
     value specified by gsTermDelayThreshold.”
     ::= { rcbsEntry 27 }

    rcbsTermMaxCount       OBJECT-TYPE
     SYNTAX Counter
     ACCESS read-only
     STATUS mandatory
     DESCRIPTION
     ”The total number of transactions which exceeded the
     value specified by gsTermDelayMax.”
     ::= { rcbsEntry 28 }

    rcbsTermDelayTraceOn   OBJECT-TYPE
     SYNTAX INTEGER { true(1), false(2) }
     ACCESS read-write
     STATUS mandatory
     DESCRIPTION
     ”Terminal delay tracing is enabled for the terminal
     defined by rcbsTerminal if rcbsTermDelayTraceOn is
     set to a non-zero value. The results of the trace
     will be found in the tdTable. For this to work,
     global terminal delay trace must be turned off — to
     do this setgsTermDelayTraceOn to 0 (this value is
     found in the gsTable).”
     ::= { rcbsEntry 29 }

    rcbsHostAlias          OBJECT-TYPE
     SYNTAX DisplayString (SIZE (0..16))
     ACCESS read-only
     STATUS mandatory
     DESCRIPTION
     ”The alias of the host computer associated with the
     RCB.”
     ::= { rcbsEntry 30 }

hd           OBJECT IDENTIFIER ::= { gwSession 3 }

    -- The HD Table

    hdTableReset     OBJECT-TYPE
     SYNTAX INTEGER { true(1), false(2) }
     ACCESS read-write
     STATUS mandatory
     DESCRIPTION
     ”Reset variable for the host delay table”
     ::= { hd 2 }
```

```
hdTableLastReset OBJECT-TYPE
 SYNTAX TimeTicks
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”The time elapsed since the last reset of the host
 delay table”
 ::= { hd 3 }

-- Table Definition

hdTable          OBJECT-TYPE
 SYNTAX SEQUENCE OF HdEntry
 ACCESS not-accessible
 STATUS mandatory
 DESCRIPTION
 ”Host delay table. Host delay is defined as the time
 from when a terminal message is received until the
 host replys. Each entry represents a bucket in the
 host delay table.”
 ::= { hd 4 }

-- Row Definition

hdEntry          OBJECT-TYPE
 SYNTAX HdEntry
 ACCESS not-accessible
 STATUS mandatory
 INDEX { hdBucket }
 ::= { hdTable 1 }

-- Columnar Object Definition

HdEntry ::=
 SEQUENCE {
   hdHostDelay INTEGER,
   hdCount     Counter,
   hdBucket    INTEGER
}

-- Leaf Definition

hdHostDelay      OBJECT-TYPE
 SYNTAX INTEGER
 ACCESS read-write
 STATUS mandatory
 DESCRIPTION
 ”The host delay time. The value in each row defines a
 host delay bucket bounded by the value in the
 preceeding row (or 0) and the value in the row. Delay
 times are defined in 1/100th seconds.”
 ::= { hdEntry 1 }
```

```
    hdCount          OBJECT-TYPE
     SYNTAX Counter
     ACCESS read-only --read-reset
     STATUS mandatory
     DESCRIPTION
     ”The total number of times the host responded to a
     message from the terminal in a time bounded by the
     associated host delay time.”
     ::= { hdEntry 2 }

    hdBucket         OBJECT-TYPE
     SYNTAX INTEGER (1..20)
     ACCESS read-only
     STATUS mandatory
     DESCRIPTION
     ”The bucket number used to identify a bucket
     in the host delay table (hdTable).”
     ::= { hdEntry 3 }

td         OBJECT IDENTIFIER ::= { gwSession 4 }

    -- The TD Table

    tdTableReset     OBJECT-TYPE
     SYNTAX INTEGER { true(1), false(2) }
     ACCESS read-write
     STATUS mandatory
     DESCRIPTION
     ”Reset variable for the tdTable”
     ::= { td 2 }

    tdTableLastReset OBJECT-TYPE
     SYNTAX TimeTicks
     ACCESS read-only
     STATUS mandatory
     DESCRIPTION
     ”The time elapsed since last reset of tdTable”
     ::= { td 3 }

    -- Table Definition

    tdTable          OBJECT-TYPE
     SYNTAX SEQUENCE OF TdEntry
     ACCESS not-accessible
     STATUS mandatory
     DESCRIPTION
     ”Terminal delay table. Terminal delay is the time
     used to send a message from a terminal to a host plus
     the time used to send a response. Host processing and
     queueing time is not included. Each entry in the
     table contains a response time bucket and an
     associated count. Set gsTermDelayTraceOn=1 to enable
     for all terminals. Set rcbsTermDelayTraceOn=1 and
     gsTermDelayTraceOn=0 to enable for single terminal.”
      ::= { td 4 }
```

```
-- Row Definition

tdEntry          OBJECT-TYPE
 SYNTAX TdEntry
 ACCESS not-accessible
 STATUS mandatory
 INDEX { tdBucket }
 ::= { tdTable 1 }

-- Columnar Object Definition

TdEntry ::=
 SEQUENCE {
    tdTrxnTime  INTEGER,
    tdCount     Counter,
    tdBucket    INTEGER
}

-- Leaf Definition

tdTrxnTime       OBJECT-TYPE
 SYNTAX INTEGER
 ACCESS read-write
 STATUS mandatory
 DESCRIPTION
 ”The round trip transaction time. The value in each
 row defines a terminal delay bucket bounded by the
 value in the preceeding row (or 0) and the value in
 the row. Delay times are defined in 1/100th seconds.”
 ::= { tdEntry 1 }

tdCount          OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only --read-reset
 STATUS mandatory
 DESCRIPTION
 ”The total number of times the terminal delay fell
 within the time interval bounded by the associated
 tdTrxnTime time.”
 ::= { tdEntry 2 }

tdBucket         OBJECT-TYPE
 SYNTAX INTEGER (1..20)
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”The bucket number used to identify a bucket in the
 terminal delay table (tdTable).”
 ::= { tdEntry 3 }
```

```
ahost          OBJECT IDENTIFIER ::= { nApplication 2 }

       ahstReset          OBJECT-TYPE
         SYNTAX INTEGER { true(1), false(2) }
         ACCESS read-write
         STATUS mandatory
         DESCRIPTION
         "Reset variable for the ahost group"
         ::= { ahost 1 }

       ahstLastReset      OBJECT-TYPE
         SYNTAX TimeTicks
         ACCESS read-only
         STATUS mandatory
         DESCRIPTION
         "The time elapsed since the last reset of ahost group"
         ::= { ahost 2 }

       ahstFramesOut      OBJECT-TYPE
         SYNTAX Counter
         ACCESS read-only
         STATUS mandatory
         DESCRIPTION
         "Number of frames from the gateway sent to the host."
         ::= { ahost 3 }

       ahstFramesIn       OBJECT-TYPE
         SYNTAX Counter
         ACCESS read-only
         STATUS mandatory
         DESCRIPTION
         "Number of frames received from the host."
         ::= { ahost 4 }

       ahstIdleDetected   OBJECT-TYPE
         SYNTAX Counter
         ACCESS read-only
         STATUS mandatory
         DESCRIPTION
         "The number of times that the host went inactive. The
         time out is set in the gateway from the user interface."
         ::= { ahost 5 }

       ahstNoErrors       OBJECT-TYPE
         SYNTAX Counter
         ACCESS read-only
         STATUS mandatory
         DESCRIPTION
         "The number of host interface errors detected by the
         gateway."
         ::= { ahost 6 }
```

```
ahstSyntaxErrors    OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Number of invalid commands received.  Incremented
  whenever a ?1 error is sent to the host.”
  ::= { ahost 7 }

ahstLengthErrors    OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Number of host frames that are too long.  Increments
  whenever a ?4 is is issued by the gateway.”
  ::= { ahost 8 }

ahstDisabledErrors  OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Number of commands sent to disabled terminals.
  Increments whenever the gateway issues a ?5 response.”
  ::= { ahost 9 }

ahstGapErrors       OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Number of times the Gap time out is exceeded.   The
  time out value is set with the CMT5 command and is
  measured by the start of a command and the receipt of a
  carriage return <CR>.  Increments whenever the gateway
  issues a ?6 response.”
  ::= { ahost 10 }

ahstCheckErrors     OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Number of times the host command fails the LRC or
  CRC-16 check.  LRC/CRC error checking is enabled by the
  CMT8 command.  Increments whenever the contoller sends a
  ?8 response.”
  ::= { ahost 11 }
```

```
ahstNoBuffersErrors OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Number of times the gateway can not receive a command
  from the host because of the lack of local buffers.
  Increments when the gateway sends a ?10 response.”
  ::= { ahost 12 }

ahstSequenceErrors  OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Number of times the host sends an R command before the
  host initialized the gateway with CMT commands.
  Increments when the gateway sends a ?11 response.”
  ::= { ahost 13 }

-- ahstOtherErrors { ahost 14 }
-- is obsolete and has been deleted.

ahstParityErrors    OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Number of async parity errors”
  ::= { ahost 15 }

ahstFrameErrors     OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Number of async character framing errors”
  ::= { ahost 16 }

ahstNoiseErrors     OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
  ”Number of UART signal samples with ambiguous results”
  ::= { ahost 17 }
```

```
         ahstBreakErrors     OBJECT-TYPE
           SYNTAX Counter
           ACCESS read-only
           STATUS mandatory
           DESCRIPTION
           ”Number of async break characters received”
           ::= { ahost 18 }

tgate          OBJECT IDENTIFIER ::= { nApplication 3    }
```

(Applies only to the 6910 Telnet Gateway/Access Point)

```
         -- The tcp connection Table

         -- Table Definition

         tgateTable          OBJECT-TYPE
          SYNTAX SEQUENCE OF TgateEntry
          ACCESS not-accessible
          STATUS mandatory
          DESCRIPTION
          ”SESSION control block status table.  Each entry
          corresponds to a terminal session.”
          ::= { tgate 3 }

         -- Row Definition

         tgateEntry          OBJECT-TYPE
          SYNTAX TgateEntry
          ACCESS not-accessible
          STATUS mandatory
           INDEX { tgateIndex }
           ::= { tgateTable 1 }

         -- Columnar Object Definition

         TgateEntry ::=
           SEQUENCE {
              tgateIndex         INTEGER,
              tgateTerminal      INTEGER,
              tgateHostIpAddress IpAddress,
              tgateHostPort      INTEGER,
              tgateGatePort      INTEGER,
              tgateEvent         INTEGER,
              tgateEventAge      TimeTicks,
              tgateIdleTime      TimeTicks,
              tgateOfflineTime   TimeTicks
           }

         -- Leaf Definition

         tgateIndex          OBJECT-TYPE
          SYNTAX INTEGER (1..65535)
          ACCESS read-only
          STATUS mandatory
          DESCRIPTION
          ”Non zero table index.”
          ::= { tgateEntry 1 }
```

```
tgateTerminal       OBJECT-TYPE
 SYNTAX INTEGER (0..127)
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”Terminal session identifier”
 ::= { tgateEntry 2 }

tgateHostIpAddress  OBJECT-TYPE
 SYNTAX  IpAddress
 ACCESS  read-only
 STATUS  mandatory
 DESCRIPTION
 ”The IP Address of host.”
 ::= { tgateEntry 3 }

tgateHostPort       OBJECT-TYPE
 SYNTAX INTEGER (0..65535)
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”The TCP port number on the host.”
 ::= { tgateEntry 4 }

tgateGatePort       OBJECT-TYPE
 SYNTAX INTEGER (0..65535)
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 ”The TCP port number on this gateway.”
 ::= { tgateEntry 5 }
```

```
tgateEvent          OBJECT-TYPE
 SYNTAX INTEGER {
                          rfClose(100),
                        rfOffline(101),
                         rfOnline(102),
                        rfTimeout(103),
                           rfSent(104),
                      hostOffline(201),
                       hostOnline(202),
                      hostTimeout(203),
                         hostSent(204)
                }
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 "Last event that occurred;
     RfClose     - Rf was asked to disconnect,
     RfOffline   - Rf disconnected.
     RfOnline    - Rf reconnected.
     RfTimeout   - Idle timeout occurred while rf
                   disconnected (Host connection
                   was dropped),
     RfSent      - Rf sent data to Host,
     HostOffline - Host connection was lost,
     HostOnline  - Host connection was
     established.
     HostTimeout - Idle timeout occurred while rf
                   connected (Host connection was
                   dropped),
     HostSent    - Host sent data to Rf."
 ::= { tgateEntry 6 }

tgateEventAge       OBJECT-TYPE
 SYNTAX TimeTicks
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 "Age of last tgateEvent."
 ::= { tgateEntry 7 }

tgateIdleTime       OBJECT-TYPE
 SYNTAX TimeTicks
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 "How much longer will this connection remain connected
 before it is disconnected due to inactivity.  Zero if
 timer is disabled.  Updated once per minute."
 ::= { tgateEntry 8 }
```

```
              tgateOfflineTime    OBJECT-TYPE
               SYNTAX TimeTicks
               ACCESS read-only
               STATUS mandatory
               DESCRIPTION
               ”The Rf connection has been lost.  How much longer will
               the host connection be maintained before it is
               disconnected.  Zero if timer is disabled.  Updated once
               per minute.”
               ::= { tgateEntry 9 }
nControl               OBJECT IDENTIFIER ::= { norandNET 105 }
      powerUp              OBJECT IDENTIFIER ::= { nControl 1 }
          pwrPowerUpCount    OBJECT-TYPE
           SYNTAX Counter
           ACCESS read-only
           STATUS mandatory
           DESCRIPTION
           ”Power-up count”
           ::= { powerUp 1 }
          pwrNextPowerUpTime OBJECT-TYPE
           SYNTAX TimeTicks
           ACCESS read-write
           STATUS mandatory
           DESCRIPTION
           ”Next power-up time (Used to reboot the device)”
           ::= { powerUp 2 }
      softwareDownLoad   OBJECT IDENTIFIER ::= { nControl 2 }
          sdStartTime         OBJECT-TYPE
           SYNTAX TimeTicks
           ACCESS read-write
           STATUS mandatory
           DESCRIPTION
           ”The amount of time to delay before beginning the software
           download”
           ::= { softwareDownLoad 1 }
          sdServerIpAddress   OBJECT-TYPE
           SYNTAX IpAddress
           ACCESS read-write
           STATUS mandatory
           DESCRIPTION
           ”TFTP server IP address”
           ::= { softwareDownLoad 2 }
          sdScriptFilename    OBJECT-TYPE
           SYNTAX DisplayString (SIZE (0..80))
           ACCESS read-write
           STATUS mandatory
           DESCRIPTION
           ”Download script filename (May include path)”
           ::= { softwareDownLoad 3 }
```

```
sdStatus            OBJECT-TYPE
 SYNTAX INTEGER {
                     sdPending(1),
                   sdSTStopped(2),
                   sdInProcess(3),
                  sdTerminated(4),
                     sdSuccess(5),
                       sdError(6),
                       pwrNPUT(7),
                     tftpError(8)
                }
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 "Status of the current software download"
 ::= { softwareDownLoad 4 }

sdErrorString       OBJECT-TYPE
 SYNTAX DisplayString (SIZE (0..40))
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 "Description of sdStatus field"
 ::= { softwareDownLoad 5 }

sdCheckPoint        OBJECT-TYPE
 SYNTAX INTEGER
 ACCESS read-write
 STATUS mandatory
 DESCRIPTION
 "An application variable intended to contain a number
 relating the progress of the current software download"
 ::= { softwareDownLoad 6 }

sdSetActivePointers OBJECT-TYPE
 SYNTAX INTEGER {
                   none(1),
                   boot(2),
                   data(3),
                   both(4)
                }
 ACCESS read-write
 STATUS mandatory
 DESCRIPTION
 "If the device reboots due to the expiration of the
 pwrNextPwrUpTime timer, this value specifies which
 active pointers will be toggled prior to rebooting"
 ::= { softwareDownLoad 7 }
```

```
sdTerminate         OBJECT-TYPE
 SYNTAX INTEGER {
                  true(1),
                  false(2)
                }
 ACCESS read-write
 STATUS mandatory
 DESCRIPTION
 ”Terminate the current software download”
 ::= { softwareDownLoad 8 }
```

END

INDEX

6910 Integrated Gateway/Access Point

INDEX

6910 Telnet Gateway/Access Point